Natural Hand Interaction for Augmented Reality

A thesis
submitted in partial fulfilment
of the requirements for the Degree
of
Doctor of Philosophy
in the
University of Canterbury
by
Thammathip Piumsomboon

Supervision and Examination Committee

Professor Andy Cockburn	Senior Supervisor
Professor Mark Billinghurst	Co-Supervisor
Dr. Adrian Clark	Co-Supervisor
Associate Professor Beryl Plimmer	Internal Examiner
Associate Professor Stephan Lukosch	External Examiner

Department of Computer Science and Software Engineering
University of Canterbury
2015

Abstract

Despite the increasing prevalence of Augmented Reality (AR) interfaces, there is still a lack of interaction techniques that allow full utilization of the medium. Natural hand interaction has the potential to offer these affordances however, as of yet, has not been well explored. The aim of this thesis is to improve the understanding of natural hand interaction and ultimately create a novel natural hand interaction technique that enhances user experience when interacting with AR.

To better understand natural hand interaction, two prototype AR systems featuring environmental awareness and physical simulation were developed, one featuring interaction on a tabletop, and the other in a mobile tablet setting. Observations and feedback from public demonstrations of the systems were collected, and it was found that users felt that interacting physically using their hands and other tangible objects was natural and intuitive. Following this, a guessability study was conducted to elicit hand gestures for AR and obtain qualitative feedback from users in a video-see through head mounted display (HMD). From the results, a user-defined gesture set was created to guide the design of natural hand interaction for AR.

Utilizing this deeper understanding and set of design guidelines, a gesture interface was developed that enabled hand tracking and gesture recognition based on depth sensing input. An AR framework that supports natural interaction as the primary input, called G-SIAR, was created, and a novel direct manipulation natural hand interaction technique, Grasp-Shell (G-Shell), was developed. This interaction technique was validated by comparing it to a traditional indirect manipulation gesture and speech interaction technique, Gesture-Speech (G-Speech), in a usability study. From the study, we gained insights into the strengths and weaknesses of each interaction technique. We found impacts on performance, usability, and user

preference when comparing G-Shell's direct interaction, where the user physically manipulates the object they are interacting with, and G-Speech's indirect interaction, where the user interacts with the object remotely using gestures and speech commands, depending on the task. We concluded that these interaction techniques were complementing each other and should be offered together.

The primary contributions of this thesis include a literature review of AR and its interaction techniques, the implementation of two AR systems and findings from the public demonstrations, findings from a guessability study on hand gestures for AR, the design and development of gesture interface and multimodal AR framework, and the design and evaluation of two natural interaction techniques, G-Shell and G-Speech. This research offers knowledge gained into natural hand interaction for AR and forms a new layer of foundation for research into interaction techniques in AR.

UNIVERSITY OF CANTERBURY
Te Whare Wānanga o Waitaha
CHRISTCHURCH NEW ZEALAND

Deputy Vice-Chancellor's Office
Postgraduate Office

Co-Authorship Form

This form is to accompany the submission of any thesis that contains research reported in co-authored work that has been published, accepted for publication, or submitted for publication. A copy of this form should be included for each co-authored work that is included in the thesis. Completed forms should be included at the front (after the thesis abstract) of each copy of the thesis submitted for examination and library deposit.

Please indicate the chapter/section/pages of this thesis that are extracted from co-authored work and provide details of the publication or submission from the extract comes:

Chapter 4 / Section 4.2 / Pages 73 - 91

(1) Piumsomboon, T., Clark, A. and Billinghurst, M. Physically-based Interaction for Tabletop Augmented Reality Using a Depth-sensing Camera for Environment Mapping. In Proc. Image and Vision Computing New Zealand (IVCNZ'11), pages 161-166, 2011. (Long)

(2) Piumsomboon, T., Clark, A., Umakatsu, A. and Billinghurst, M. Physically-based natural hand and tangible AR interaction for face-to-face collaboration on a tabletop. In IEEE Symposium on 3D User Interfaces (3DUI'12), pages 155-156, 2012. (Poster)

Chapter 4 / Section 4.3 / Pages 91 - 98

(3) Piumsomboon, T., Clark, A. and Billinghurst, M. KITE: Platform for Mobile Augmented Reality Gaming and Interaction using Magnetic Tracking and Depth Sensing. In 12th IEEE International Symposium on Mixed and Augmented Reality (ISMAR'13), 2013. In press. (Poster & Demo)

Chapter 5 / Section 5.1 to 5.5 / Pages 103 - 128

(4) Piumsomboon, T., Clark, A., Billinghurst, M. and Cockburn, A. User-Defined Gestures for Augmented Reality. In Human-Computer Interaction – INTERACT 2013, P. Kotzé, G. Marsden, G. Lindgaard, J. Wesson and M. Winckler, Eds. Springer Berlin Heidelberg, 282-299, 2013. (Long)

(5) Piumsomboon, T., Clark, A., Billinghurst, M. and Cockburn, A. User-defined gestures for augmented reality. In Proceedings of the CHI '13 Extended Abstracts on Human Factors in Computing Systems, Paris, France, 955-960, 2013. (Work-in-Progress)

Chapter 7 / Section 7.1 – 7.4 / Pages 155 – 178 AND Chapter 8 / Section 8.1 – 8.2 / Pages 181 - 196

(6) Piumsomboon, T., Altimira, D., Kim, H., Clark, A., Lee, G. and Billinghurst, M. Grasp-Shell vs Gesture-Speech: A comparison of direct and indirect natural interaction techniques in Augmented Reality. In 13th IEEE International Symposium on Mixed and Augmented Reality (ISMAR'14), 2014. (Long)

(7) Piumsomboon, T., Clark, A. and Billinghurst, M. G-SIAR: Gesture-Speech Interface for Augmented Reality. In 13th IEEE International Symposium on Mixed and Augmented Reality (ISMAR'14), 2014. (Demo)

Please detail the nature and extent (%) of contribution by the candidate:

Publications are ordered by numbers in the previous section.

Publication (1): 50% - Co-developed the system and co-authored the paper.

Publication (2): 90% - Co-developed the system and co-authored the paper.

Publication (3): 90% - Developed the system and co-authored the paper.

Publication (4): 90% - Developed the system, conducted the study, and co-authored the paper.

Publication (5): 90% - Developed the system, conducted the study, and co-authored the paper.

Publication (6): 85% - Developed the system, conducted the study, and co-authored the paper.

Publication (7): 90% - Developed the system and co-authored the paper.

Certification by Co-authors:

If there is more than one co-author then a single co-author can sign on behalf of all
The undersigned certifys that:

- The above statement correctly reflects the nature and extent of the PhD candidate's contribution to this co-authored work
- In cases where the candidate was the lead author of the co-authored work he or she wrote the text

Name: ***Adrian Clark*** Signature:  Date: ***27/02/15***

Publications from this Thesis

The peer-reviewed publications listed below are based on research completed as part of this thesis.

1. **Piumsomboon, T.**, Clark, A. and Billinghurst, M. Physically-based Interaction for Tabletop Augmented Reality Using a Depth-sensing Camera for Environment Mapping. In Proc. Image and Vision Computing New Zealand (IVCNZ'11), pp. 161-166, 2011. (Chapter 4)

2. **Piumsomboon, T.**, Clark, A., Umakatsu, A. and Billinghurst, M. Physically-based natural hand and tangible AR interaction for face-to-face collaboration on a tabletop. In IEEE Symposium on 3D User Interfaces (3DUI'12), pp. 155-156, 2012. (Chapter 4)

3. **Piumsomboon, T.**, Clark, A. and Billinghurst, M. KITE: Platform for Mobile Augmented Reality Gaming and Interaction using Magnetic Tracking and Depth Sensing. In 12th IEEE International Symposium on Mixed and Augmented Reality (ISMAR'13), pp. 289-290, 2013. (Chapter 4)

4. **Piumsomboon, T.**, Clark, A., Billinghurst, M. and Cockburn, A. User-Defined Gestures for Augmented Reality. In Human-Computer Interaction – INTERACT 2013, P. Kotzé, G. Marsden, G. Lindgaard, J. Wesson and M. Winckler, Eds. Springer Berlin Heidelberg, pp. 282-299, 2013. (Chapter 5)

5. **Piumsomboon, T.**, Clark, A., Billinghurst, M. and Cockburn, A. User-defined gestures for augmented reality. In Proceedings of the CHI '13 Extended Abstracts on Human Factors in Computing Systems, Paris, France, 955-960, 2013. (Chapter 5)

6. **Piumsomboon, T.**, Altimira, D., Kim, H., Clark, A., Lee, G. and Billinghurst, M. Grasp-Shell vs Gesture-Speech: A comparison of direct and indirect natural interaction techniques in Augmented Reality. In 13th IEEE International Symposium on Mixed and Augmented Reality (ISMAR'14), pp. 73-82, 2014. (Chapter 7 and 8)

7. **Piumsomboon, T.**, Clark, A. and Billinghurst, M. G-SIAR: Gesture-Speech Interface for Augmented Reality. In 13th IEEE International Symposium on Mixed and Augmented Reality (ISMAR'14), pp. 365-366, 2014. (Chapter 7 and 8)

During my doctoral studies, the following papers were also published, but are not part of this thesis.

8. Clark, A., **Piumsomboon, T.** and Billinghurst, M. Augmented reality micromachines. In SIGGRAPH Asia 2011 Emerging Technologies (SA'11), pp. 1, 2011.

9. Clark, A. and **Piumsomboon, T**. A realistic augmented reality racing game using a depth-sensing camera. In Proceedings of the 10th International Conference on Virtual Reality Continuum and Its Applications in Industry (VRCAI '11), pp. 499-502, 2011.

10. Clark, A., Dünser, A., Billinghurst, M., **Piumsomboon, T.** and Altimira, D. Seamless interaction in space. In Proceedings of the 23rd Australian Computer-Human Interaction Conference (OzCHI '11), pp. 88-97, 2011.

11. **Piumsomboon, T.** and Clark, A. ARMicroMachines: natural interaction with augmented reality scenes. In Proceedings of the 13th International Conference of the NZ Chapter of the ACM's Special

Interest Group on Human-Computer Interaction (CHINZ'12), pp. 99-99, 2012.

12. **Piumsomboon, T.**, Clifford, R., and Bartneck, C. Demonstrating Maori Haka with kinect and nao robots. In ACM/IEEE international conference on Human-Robot Interaction (HRI'12), pp. 429-430, 2012.

13. Allen, M., Hoermann, S., **Piumsomboon, T.** and Regenbrecht, H. Visual Occlusion in an Augmented Reality Post-Stroke Therapy Scenario. In Proceedings of the 14th International Conference of the NZ Chapter of the ACM's Special Interest Group on Human-Computer Interaction (CHINZ'13), 2013.

14. Billinghurst, M., **Piumsomboon, T.** and Bai, H. Hands in Space: Gesture Interaction with Augmented-Reality Interfaces. IEEE Computer Graphics and Applications, vol. 34, no. 1, pp. 77-80, 2014.

Acknowledgments

I wish to express sincere appreciation to the following people for their advice and support throughout my research. Without them this thesis would not be possible.

Firstly, I would like to express my deepest gratitude to all of my supervisors. Great thanks to Professor Mark Billinghurst, for welcoming me into HITLab NZ, giving me the opportunity and support in pursuing my Ph.D., and conducting research under his guidance. His optimism, enthusiasm and hard-working attitude toward research have set a great example for me. Without him, I would not be as motivated as I am. Also great thanks to Professor Andy Cockburn, who kindly accepted me under his supervision. His valuable lessons in a rigorous approach toward research in Human-Computer Interaction will remain with me for all time. Without his advice, I would have been lost and far from achieving my goal. Last but not least, many thanks to Dr. Adrian Clark, who I consider a mentor, a colleague, and a friend. He provides constant help and advice throughout my Ph.D. Without him, this thesis would be far from completion.

Thank you to my colleagues in the HITLab NZ who helped me with my publications (Gun, David, and Hyungon) along with other current and past HITLab NZ colleagues and staffs who have provided me with their friendship and support.

Lastly, I would like to dedicate this work to my parents, who always believed in me. This would not have been possible without your unwavering support.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Our hands allow us to physically manipulate objects and perform complex tasks such as using tools. In terms of human-computer interaction, hands also play a crucial role in operating computer input devices, from a mouse and keyboard to touch input on a touchpad and surface computing. The use of hands as a means for interaction is likely to continue for future interfaces, such as Augmented Reality (AR) and Virtual Reality (VR), where three-dimensional (3D) interaction can be performed through natural hand interaction. Of particular interest, as AR overlays virtual content onto the real world, natural hand interaction allows users to perform tasks in both the real and virtual environment at the same time (Azuma, 1997), providing a natural and intuitive way to enable users to interact between the two worlds, bridging them into one seamless realm.

With the introduction of the Microsoft Kinect in 2011, depth sensing technology for natural interaction was finally available at a price affordable to consumers. The Kinect, and the consumer level depth sensors which followed, have proven to be crucial hardware for allowing the computer to better understand the environment it is operating in (Newcombe et al., 2011), in addition to facilitating body movement as an input (Shotton et al., 2011). The convergence of depth sensing technologies and hand tracking algorithms has resulted in better hand tracking software. High degree of freedom (*dof*) hand pose estimation is now achievable without the need for data gloves or external sensing (Wang et al., 2011). This allows for designing more complex natural hand interaction than could be achieved in past research (Fernandes & Fernandez, 2009; Heidemann et al., 2004; Kaiser et al., 2003; Mathias Kolsch et al., 2006; Lee & Hollerer, 2007). Because of these limitations in prior

research, hand gestures were considered only as an auxiliary input to speech. As a result, little is known about the benefits of natural hand interaction and user preference when interacting in AR.

Despite the increasing prevalence of AR interfaces, there is still a lack of interaction techniques that allow full utilization of the medium. Natural hand interaction has the potential to offer these affordances, and has not been well explored. It is necessary to further study natural hand interaction to learn its potentials and limitations, and empirical evidence must be gathered from formal evaluation to validate our beliefs. Therefore, the focus of this thesis is on *understanding and enhancing natural hand interaction for AR*, where natural hand interaction is the primary modality compared to the traditional gesture and speech multimodal input, which serves as a baseline interaction technique in this research.

This thesis's objective is composed of two parts: to gain insights into user behavior and preference and create a set of design guidelines for interaction using natural hand gestures in various AR settings; and to create a novel natural hand interaction technique for AR.

Figure 1.1 illustrates the thesis's goals and their subgoals. The first goal, *Understanding Natural Hand Interaction in AR*, is attained by a collection of observations and feedback from public demonstrations of two AR systems; one featuring interaction on a tabletop setting, and the other using a mobile tablet. A guessability study is conducted to elicit hand gestures and to gain qualitative feedback from users while using a video see-through head mounted display (HMD).

The outcome of the first goal leads to the design guidelines for the second goal, *Enhancing Natural Hand Interaction in AR*. To achieve this goal, a gesture interface is developed. It streamlines hand tracking and gesture recognition using a depth sensor. An AR platform, which uses natural hand interaction and

gesture-speech as the primary inputs, is created and a novel natural hand interaction technique is developed following the guidelines from completing the first goal. The proposed technique is then validated by comparing it to the baseline technique in gesture and speech interaction.

In the sections to follow, the research problem is stated in Section 1.1, the research approach is presented in Section 1.2, research contributions are covered in Section 1.3, and the thesis structure, and glossary of terms are discussed in Section 1.4 and 1.5, respectively.

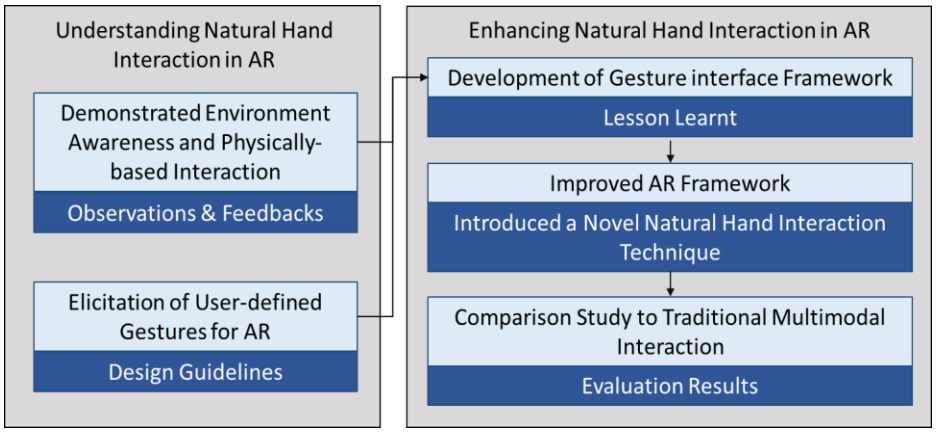


Figure 1.1: Breakdown of the thesis's objective

1.1 Problem Statement

The two primary goals of this research are to *understand* and later *enhance* natural hand interaction in AR. These two primary goals can be further divided into subgoals, which we describe in the following list together with their outcomes.

SG-1. Understand the best practices and limitations of the technology of current AR interfaces and interaction techniques. The outcome of this goal is literature reviews and guidelines from existing research.

SG-2. Learn from users through observing their interaction with an AR system that offers environment awareness and physically-based interaction, then use this information to determine the characteristics of affordance of such an AR interface. The outcome of this goal is insights into user's natural behavior and approaches that can be taken to improve and enhance user experience through natural hand interaction.

SG-3. Learn hand gestures that are preferable and easy to perform in AR from users and create design guidelines from the findings. Successful completion of this goal will result in the first set of user-defined gestures for AR and the classification of those gestures into a gesture taxonomy for AR.

SG-4. Develop a gesture interface that utilizes depth sensing technology for hand tracking and recognition. The outcome of this goal is an interface which supports novel natural hand interaction techniques.

SG-5. Develop an AR framework that supports natural interaction as the primary inputs, in this case, a direct natural hand interaction and an indirect multimodal gesture and speech interaction. The success of this goal is based on a working and demonstrable system implemented using this framework.

SG-6. Evaluate and compare two natural interaction techniques, a novel direct natural hand interaction technique and an indirect multimodal gesture and speech interaction technique. The success of this goal is based on whether there are differences between the two interaction techniques in term of performance, usability, task load, and user preference. Moreover, the strengths and weaknesses of each technique should be identified.

1.2 Research Approach

The focus of this thesis is to gain a better understanding into how natural hand interaction can improve user experience in AR.

A review was conducted on prior research in AR and relevant research area. This includes an overview of AR in terms of its definition, application domains, evaluation techniques, and usage in collaboration. Due to the paradigm shift that occurred with the arrival of consumer level depth sensors, we split our review of user interfaces and interaction techniques in AR into pre- and post- the arrival of consumer depth sensors.

Past research investigating tangible interaction in AR found that interaction through a physical object elevated the user's expectation that virtual contents should behave like their physical counterpart and so raised a question if a physics-enabled system would increase the realism and intuitiveness for the users (Hornecker & Dünser, 2009). Furthermore, with the integration of depth sensing technology, where spatial information about the scene can be obtained in real-time, systems are now capable of being aware of the physical environment (Newcombe et al., 2011). To better understand the user's natural behavior and preferences when interacting with a physics-enabled and environment aware AR system, demonstrations were given to the public and observations and feedback were gathered. In these demonstrations, natural hand interaction was introduced using basic physically-based interaction metaphors such as pushing and lifting virtual objects in the scene. The findings from the observations and feedback raised questions such as "How can natural hand interaction support more commands?" and "Can the precision of interaction be improved?"

To determine how to best design hand interaction for AR, a guessability study was conducted to elicit natural and easy to perform gestures from users

for a broad range of tasks in AR. The study yielded the first comprehensive user-defined gestures set for AR, and the gestures elicited were classified using an extended gesture taxonomy for AR.

To improve the precision of interaction and to support user-defined gestures, a gesture interface was developed. Designed to work using vision based inputs from color and depth cameras, the interface can be divided into five components: (1) the hardware interface that grabs images from the input devices, (2) the hand segmentation and tracking component, (3) the hand classification based on random forest component, (4) the hand modeling using a physics engine component, and (5) the hand posture and gesture recognition component. Through the development of this interface, valuable lessons were learnt which are included in the discussion.

Observations and feedback were obtained from extensive public demonstrations and through many iterations the framework was improved. The final outcome of this is "G-SIAR" (Gesture-Speech Interface for Augmented Reality), an AR framework that supports natural hand interaction and multimodal gesture and speech interaction as the primary inputs.

Based within this framework, two natural interaction techniques have been developed, the first, Grasp-Shell (G-Shell), is a novel natural hand direct manipulation interaction technique designed using the user-elicited gesture set, while the second, Gesture-Speech (G-Speech), is a more traditional multimodal indirect manipulation interaction technique designed using guidelines from prior research. A study was conducted to evaluate and compare the techniques in terms of task completion time, usability, task load, and preference.

1.3 Research Contributions

This research makes six main contributions to the field of natural interaction in AR. These are:

1. A literature review of interaction in Augmented Reality and related fields. The review focuses on the history of interaction in AR and looks at research before and after the introduction of depth sensing and impact this has had.
2. Two AR systems which both support environment awareness, physically-based interaction, and basic natural hand interaction. The first AR system supports face-to-face collaboration in a tabletop setting while the other supports mobile AR. The lessons learnt from the development as well as observations and feedback from public demonstrations are shared.
3. A guessability study on user-defined gestures for AR offering a number of useful outcomes including: the first comprehensive set of user-defined gestures for AR, classification of the elicited gestures based on a gesture taxonomy for AR, the agreement scores of gestures for selected tasks and their subjective rating, the qualitative findings from the design process, and the implications of this work for AR, gesture interfaces, and gesture recognition.
4. A gesture interface for AR comprising of five major components; (1) the hardware interface; (2) hand segmentation and tracking; (3) hand region classification; (4) modeling to support physics simulation; and (5) gesture recognition.
5. The G-SIAR framework that offers natural hand interaction and gesture-speech interaction as native inputs. The framework supports highly immersive and large viewing coverage of the interaction space,

transition between AR and VR, physics-enabled simulation with high accuracy, real-time and seamless object creation and interaction, and realistic rendering with shadows and hand occlusion.

6. The design and evaluation of two interaction techniques, Grasp-Shell, a natural hand interaction technique and Gesture-Speech, a multimodal gesture-speech interaction technique. The study yields insights into user performance and preferences using each technique in three manipulation tasks.

1.4 Structure of the Thesis

This thesis is composed of four parts: (i) a review of relevant research in AR and its interaction techniques; (ii) early development of natural hand interaction techniques in AR and a guessability study conducted to elicit user inputs in designing natural hand interaction; (iii) the design and implementation of a gesture interface for AR and a new AR framework together with two natural interaction techniques to improve user experience in AR and a formal evaluation to validate and compare these techniques; and (iv) the discussion of our research and possible future research directions following this thesis.

Part I gives an overview of previous research in AR and related interaction techniques. Within this, Chapter 2 discusses AR and its interaction techniques prior to the release of the first Microsoft Kinect at the end of 2010, then introduces depth sensing technology and includes more recent research in AR interaction techniques that utilizes depth information from 2011 onward. The shortcomings of previous research that lead to the proposal of our subgoals are summarized toward the end of this chapter.

Part II explores environmental awareness and physically-based interaction, and examines the gestures elicited from users for natural hand interaction in AR.

Within this, Chapter 3 describes case studies in environment awareness and physically-based interaction in two settings; (1) collaborative face-to-face interface on a tabletop setting, (2) natural hand interaction on a mobile tablet. Chapter 4 covers the result of the first guessability study for hand gestures for AR.

Part III presents the development of the gesture interface and our AR framework that supports both natural hand and gesture-speech interaction. Within this, Chapter 5 shares the implementation and the lessons learnt from the development of a gesture interface that supports hand tracking, classification, and recognition. Chapter 6 introduces the G-SIAR framework and two natural interaction techniques, Grasp-Shell and Gesture-Speech. Chapter 7 reports the results from the comparison study between the two interaction techniques.

In Part IV, the discussion of our research is covered in Chapter 8, this includes our research progress, the design guidelines from this research, the discussion of natural hand interaction for AR, and future work. This thesis concludes in Chapter 9.

Part I
Augmented Reality and Interaction Techniques

Chapter 2
Prior Work in Augmented Reality and Natural Interaction

In his paper, "The ultimate display" (Sutherland, 1965), Ivan Sutherland, considered by some to be the creator of the modern Graphical User Interface (GUI) (Sutherland, 1964), described his vision of the future of computing technology:

> *"The ultimate display would, of course, be a room within which the computer can control the existence of matter. A chair displayed in such a room would be good enough to sit in. Handcuffs displayed in such a room would be confining, and a bullet displayed in such a room would be fatal. With appropriate programming such a display could literally be the Wonderland into which Alice walked."*

Sutherland must have envisaged the endless possibilities of how our reality can be augmented and technology integrated into the physical world. The fulfillment of this vision relies on advancement of the two fundamental aspects of Human Computer Interaction (HCI): the user interface and user interaction, which govern what the user senses and how they interact, respectively. In an attempt to reach this, this thesis focuses on AR as the user interface and Natural Interaction (NI) as the user interaction.

Since Sutherland's pioneering work in 1965, there has been a significant amount of research and development in technology that enables AR and natural interaction. In this chapter, a comprehensive literature review of AR interfaces and related natural interaction is presented. Beyond this review, the remaining sections in this chapter include a brief history of augmented reality in Section 2.1, interaction techniques prior to depth sensing era in Section 2.2, an overview

of depth sensing technology and its impact on natural hand tracking and interaction in Section 2.3, the current state of the art in augmented reality with depth sensing in Section 2.4, the shortcomings in previous research in Section 2.5, and a conclusion in Section 2.6.

2.1 A Brief History of Augmented Reality

Augmented Reality (AR) describes one of all the computer interface techniques on the Reality-Virtuality Continuum that was introduced by Milgram and Kishino (Milgram & Kishino, 1994), as shown in Figure 2.1. AR overlays computer generated information onto the real world (Azuma, 1997; Feiner et al., 1993) and in doing so offers an human computer interface that is close to physical reality. Researchers and designers have envisioned AR to offer ubiquitous computing where users can interact with both virtual and real-world content at the same time, thus interfacing with the computing device without losing immersion in the surrounding environment. AR has the potential to revolutionize Human-Computer Interaction (HCI), changing the way we interface and interact with information. A compelling illustration of this idea is illustrated by Matsuda (Matsuda, 2010) as shown in Figure 2.2.

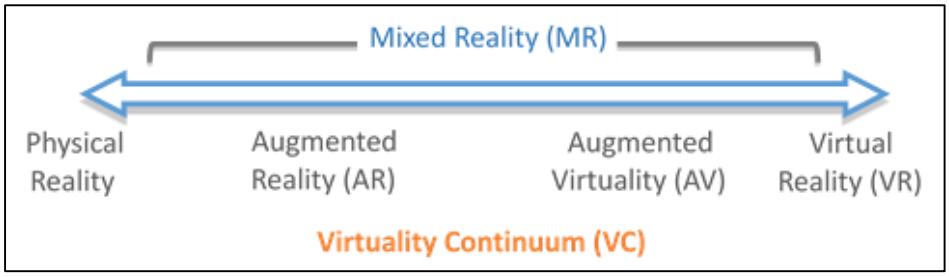


Figure 2.1: Milgrim's Virtuality Continuum (Milgram & Kishino, 1994)

Figure 2.2: Matsuda's vision of AR (Matsuda, 2010)

2.1.1 Fundamental Components, Applications, and Evaluation in Augmented Reality

2.1.1.1 Fundamental Subsystems of AR

Reicher et al. (2003) found that research prototypes in AR usually do not emphasize software architecture due to the focus on a particular task, nevertheless, common fundamental attributes for AR were found. The high-priority attributes were tracking and rendering latency, wireless and network-disconnected operation, use of multiple tracking devices, component addition and reuse, and the ability to integrate existing AR components. The low-priority

attributes were limiting CPU load, fault tolerance and system uptime, security, re-configurability at runtime, support for different simultaneous input modalities, adaptability to users' preferences and abilities, support for multiple users, support for multiple hardware platforms, and ease of integrating legacy components.

They identified six fundamental subsystems common to most AR applications; (1) *Application* contains application-specific logic and content, and access to legacy systems and databases, (2) *Tracking* is responsible for determining the users' and other objects' pose, (3) *Control* gathers and processes user input, (4) *Presentation* uses 3D and other output modalities, (5) *Context* collects different types of context data and makes it available to other subsystems, and (6) *World Model* stores and provides information about real and virtual objects around the user.

2.1.1.2 Applications in AR

A comprehensive survey by Azuma (Azuma, 1997) gave an early insight into how AR could be applied to a number of different application areas. Six major classes of application were explored including medical, manufacturing and repair, annotation and visualization, robot path planning, entertainment, and military aircraft. Later, Azuma et al. (Azuma et al., 2001) conducted another survey and regrouped the potential applications into three groups including mobile, collaborative and commercial. They stated that mobile AR could enable outdoor applications such as navigation, situational awareness, and geo-located information retrieval, while the largest commercial applications of AR would likely be advertisement. For collaborative applications, they gave examples in the area of entertainment and gaming such as AR air hockey, collaborative combat and an AR pool game.

Later on, van Krevelen and Poelman (van Krevelen & Poelman, 2010) conducted another survey where they categorized AR applications into personal information systems (e.g. personal assistance and advertisement, navigation and touring), industrial (e.g. design, assembly and maintenance), military (e.g. combat simulation), medical, entertainments (e.g. games), AR for the office, education and training. Today, AR has been utilized to assist many areas, for example crime scene investigation (Poelman et al., 2012), VIP protection, forensic investigation, and domestic violence (Datcu et al., 2014).

2.1.1.3 User Evaluation in AR

Swan and Gabbard (2005) categorized user evaluation in AR into three categories; *Perception*: How do users perceive virtual information overlaid on the real world? What perceptual cues can be used to distinguish between real and virtual content?, *Performance:* How does one interface perform against another?, and *Collaboration*: How can AR interfaces be used to enhance face-to-face and remote collaboration? They found that up to the year 2005 there was very little AR research on user-centered design.

In 2008, a more thorough study and broader sampling was conducted by Dünser et al. (2008) and they introduced the fourth category of *System Usability*. The same study showed a significant change in research trends where user performance became the main AR focus over user perception. Similarly, Billinghurst (2008) classified user studies in AR into three categories: *Perception, Interaction*: How do users interact with virtual information overlaid on the real world? How can real world objects be used to interact with augmented content?, and *Collaboration*. A common finding amongst these surveys was that AR research on collaboration contributed only a small fraction of the total number of papers published for example the 2008 survey found only 10 papers on collaboration from of a total of 161 papers published.

2.1.1.4 Collaboration in AR

A collaborative AR system can support multiple users. Billinghurst and Kato (2002) outlined the five reasons why AR interfaces are ideal for collaboration; (1) seamless interaction between real and virtual environments, (2) the ability to enhance reality, (3) the presence of spatial cues for face-to-face and remote collaboration, (4) support of a tangible interface metaphor, and (5) the ability to transition smoothly between reality and virtuality. In the remainder of this section, we present past research in face-to-face collaboration.

One of the earliest AR interfaces to support face-to-face collaboration was StudierStube (Schmalstieg et al., 1996). In this system, multiple users could wear a head mounted display (HMD) and be able to view the same 3D virtual object from their own viewpoint. However, StudierStube required the users to use special tracked input devices to interact with the virtual content, see Figure 2.3 (left).

Billinghurst et al. (1998) combined AR with the traditional computer supported collaborative work (CSCW) principles in Shared Space. This early work explored 3D manipulation of virtual images in collaborative web browsing and personal information space applications. Later, spatial and physical interactions were introduced in a Mixed Reality (MR) application (Billinghurst et al., 2000), where users could examine and manipulate virtual objects using physical cards. Neither of these applications supported natural hand interaction, or awareness of the surrounding physical environment.

Kiyokawa et al. (1998) conducted studies comparing interaction in a shared augmented environment (SAE) to a shared virtual environment (SVE) as well as developing VLEGO II, an immersive modeler application, based on SAE. They found that a SAE was more informative than a SVE due to the fact that collaborators could see each other making their communication more effective.

Ohshima et al. (1998) developed AR2Hockey, a collaborative AR game, where two users played air-hockey on a shared physical table, using real mallets and a virtual puck. Their experiments showed that participants could play the virtual game as naturally as the real game.

Butz et al. (1999) developed EMMIE, Environment Management for Multiuser Information Environments that integrated various technologies and techniques including virtual elements such as 3D widgets and physical objects such as tracked displays and input devices as shown in Figure 2.3 (right). Collaboration was achieved through interaction within the "virtual ether" a pervasive shared visualization space, which users could view and interact with through various display systems. Moreover, they also discussed privacy management and proposed an approach to manage it. Regenbrecht et al. (2002) built MagicMeeting a system that enabled manipulation of two-dimensional (2D) and 3D data via a Tangible User Interface (TUI) with seamless data exchange support between these two spaces. They implemented a scenario in automotive design as a proof of concept.

Augmented Surface (Rekimoto & Saitoh, 1999) extended a shared workspace using a projector projecting onto tables and walls to achieve a collaborative work space across multiple users and devices. Billinghurst et al.

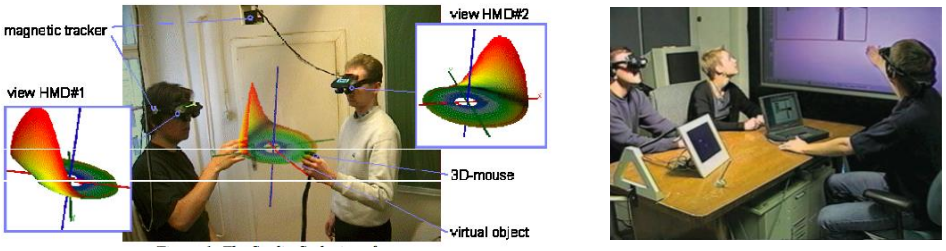


Figure 2.3: Collaboration within the StudierStube (Schmalstieg et al., 1996) (left) and EMMIE (Butz et al., 1999) (right)

(2001b) proposed using a book as a medium for collaboration where multiple users could share the digital content of the book through the fiducial markers at

the same time. Tamura et al. (2001) created another collaborative mixed reality game called AquaGaunlet where users could wear a gauntlet and interact collaboratively (see Figure 2.4 (left)).

Ishii et al. (2002) created Luminous Table for urban design and planning applications, integrating sketches, physical models, and computational simulations into a workspace. Their system tracks physical objects using cameras and uses a 2D video projection to simulate sunlight shadows, wind patterns, and traffic to augment 2D drawings and 3D physical models. In a similar application area, Broll et al. (2004) created ARTHUR, which made use of multiple frameworks to enhance architectural design and urban planning on a round table with support of TUI and basic gestures, see Figure 2.4 (right).

Benko et al. (Benko, Ishak, & Feiner, 2004) presented VITA, Visual Interaction Tool for Archaeology, using multimodal interaction (speech, touch and 3D hand gestures) for archaeologists to collaborate over a digital reconstruction of a dig site in mixed reality. Wilson and Benko (Andrew D. Wilson & Benko, 2010) introduced LightSpace, which used multiple depth cameras and projectors to create an interactive environment which spread between a table, a wall and in mid-air between them. Depth cameras monitored users' action so that they could pick up or drop the virtual objects from one

Figure 2.4: Collaborative AR game in AquaGaunlet (Tamura, Yamamoto, & Katayama, 2001) (left) and urban planning and industrial design in ARTHUR (Broll et al., 2004) (right)

surface on to another using their bare hands. However, their system focuses on interaction with virtual 2D objects such as photos and videos.

2.1.2 Early User Interfaces in Augmented Reality

In this section, we begin with a brief look into the evolution of user interfaces that influenced this research in Section 2.1.2.1. Intelligent user interfaces in AR are covered in Section 2.1.2.2 and the two main classes of these adaptive user interfaces and multimodal user interfaces are discussed in more detail in Section 2.1.2.3 and 2.1.2.4, respectively.

2.1.2.1 User Interfaces: Direct Manipulation vs Intelligent Agent

Sutherland's Sketchpad introduced two groundbreaking concepts, the Graphical User Interface (GUI) and Direct Manipulation (DM) (Sutherland, 1964). Later, Engelbart and English created a DM device which became known as the first mouse (Engelbart & English, 1968). These creations paved the way for Human Computer Interface research, and coined the frequently used term, "user interface", which is defined as the space where interaction between humans and machines occurs. Following the success of the WIMP (Windows, Icons, Menus, Pointers) model, the WYSIWYG (What You See Is What You Get) model became the leading principle of interface design.

User interface and interaction design are coupled so tightly that difficulties arise when attempting to introduce a new mode of interaction to an existing interface. For example, in a WIMP interface the mouse movement on the surface of a table is intuitively translated into the movement of the pointer on the screen allowing fast and efficient interaction. However, if we replace the input medium from a mouse to speech input instead, there is no obvious efficient solution to handle the same task. In one example attempt, Microsoft Windows 7 allows the user to select an area of the screen using speech by partitioning the screen into numbered cells that user can choose and dividing the area recursively until the desired location is reached.

Nevertheless, if the UI is intelligent and capable of interpreting a user's natural speech as one would expect from a person, the intelligent agent can save a lot of user time and proactively complete the user desired task with just a sentence or command. This raised a debate on future UI design directions between Shneiderman and Maes (1997) over DM and Interface Agents (IA). A compromised approach called the Mixed Initiative (MI) was later proposed (Horvitz, 1999), where both interfaces coexist simultaneously, allowing the user to decide whether to take the complete control or delegate tasks to the machine.

Currently, Artificial Intelligence (AI) techniques such as machine vision and learning are increasingly important in the development of modern user interfaces. At present, computer systems are accessible to users of all ages and training. *Adaptive* and *multimodal user interfaces* are promising solutions that do not assume a one-size-fits-all model. When the interface is capable of learning about its users and their preferences as well as allowing for multiple modes of interaction, it is often referred to as being *intelligent.*

2.1.2.2 Overview of Intelligent User Interfaces (IUIs) in AR

Maybury and Wahlster (1998) define *Intelligent User Interfaces (IUIs)* as human-machine interfaces that aim to improve the efficiency, effectiveness, and naturalness of human-machine interaction by representing, reasoning, and acting on models of the user, domain, task, discourse, and media (e.g., graphics, natural language, gesture). As a consequence, this interdisciplinary area draws upon research at the intersection of HCI, ergonomics, cognitive science, and

Artificial Intelligence (AI), as illustrated in Figure 2.5, with the general IUI architecture shown in Figure 2.6.

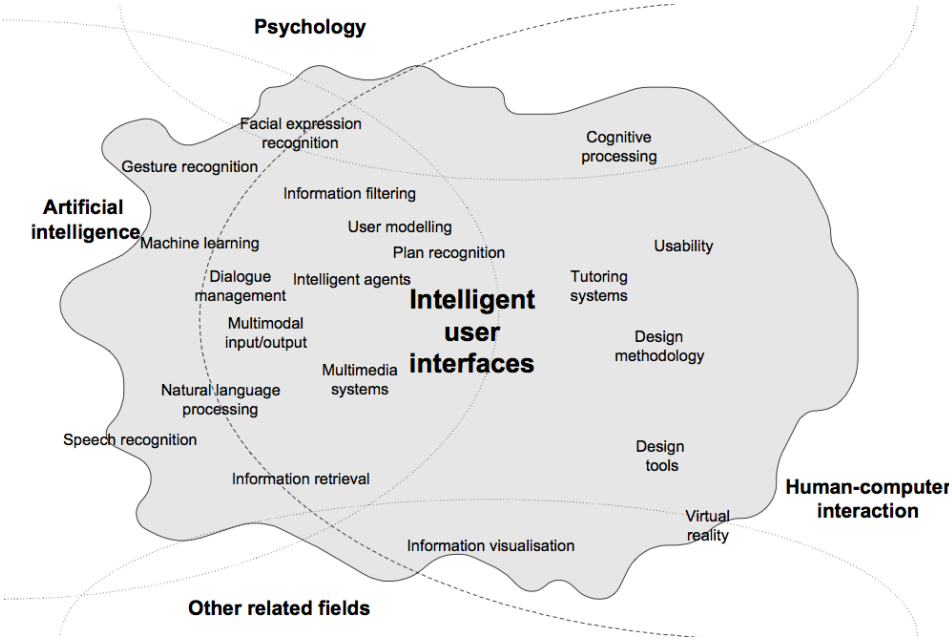


Figure 2.5: IUI research field and example of its topics (Maybury & Wahlster, 1998)

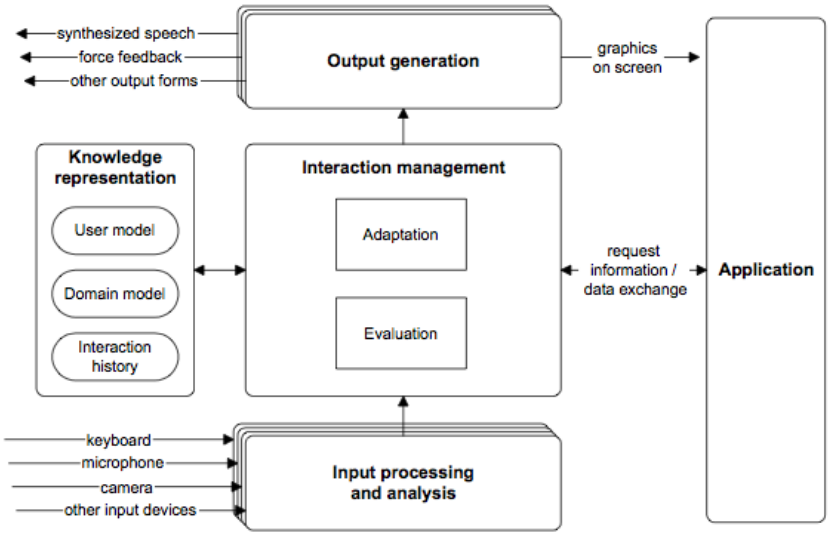


Figure 2.6: General IUI architecture (Maybury & Wahlster, 1998)

Ross (2000) has further divided IUIs into three classes: adaptation within DM interfaces, intermediary interfaces, and agent interfaces. For *adaptation within DM interfaces*, intelligent components act as an intermediary between the user and the DM interface. Example tasks include filtering information, generating suggested data and adapting visualization techniques. The *intermediary interfaces* offer users an agent that gives advice on their actions, for example making suggestions, correcting misconceptions and guiding users through tasks. *Agent interfaces* are an autonomous system that proactively delegates user tasks, in addition to making suggestions, the system can take action by itself similar to the way that programming by demonstration systems operates.

This research focuses on applying the first category of IUIs, *adaptation within DM interfaces*, to AR. Current IUI techniques that fall into this category are *adaptive user interface* and *multimodal user interfaces. Adaptive user interface (AUI)* are a composition of user modeling techniques (e.g. inferring knowledge about a user based on direct observation or posed questions) and the ability of context awareness. *Multimodal user interfaces (MUI)* are a combination of natural inputs that allows multiple modes of interaction (e.g. gesture tracking and recognition, gaze tracking, natural language processing and lip reading) and smart outputs that generate and synthesize the result in an intuitive way (e.g. information visualization and tactile feedback).

Bonanni et al. (2005) presented a framework for designing an IUI that informed and choreographed multiple tasks in a single space according to a model of tasks and users. As a proof of concept, they constructed an AR kitchen (see Figure 2.7), which had been outfitted with systems to gather data from tools and surfaces and project multi-modal interfaces back onto the tools and surfaces themselves. Their IUI was designed to inform and choreograph the tasks of preparing food to make the process more easy, safe and efficient. Based on the

user performance, they adaptively modulate the interface so that it does not intrude the user unnecessarily. Their AR IUI system can tailor information modalities based on the spatial and temporal qualities of the task, and the expertise, location and progress of the user and also choreograph multiple tasks in the same space at the same time.

Figure 2.7: AR Kitchen (Bonanni, Lee, & Selker, 2005), information projection on (1) refrigerator, (2) range, (3) cabinet, (4) faucet, and (5) drawers

Barakonyi and Schmalstieg (2007, 2008) have created an intermediary interface for AR. At the crossroad between ubiquitous computing (Ubicomp) (Weiser, 1993), interface agents (Maes & Kozierok, 1993) and AR, they introduced a system of IUI which used an animated agent, called UbiAgent, as a communication avatar. Their intelligent system observed and learned the user preference of interface appearance, then stored the parameters in a profile. The system architecture and illustrations of their domain application, LEGO Robot Maintenance, is illustrated in Figure 2.8.

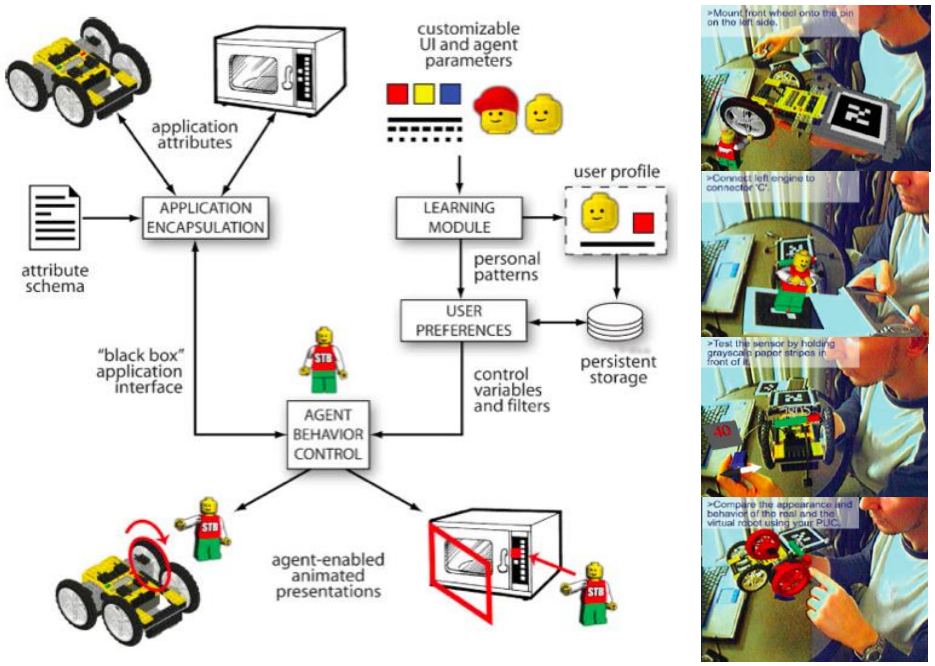


Figure 2.8: Application encapsulation with schema and AUI personalization in LEGO Robot Maintenance application (Barakonyi & Schmalstieg, 2007, 2008)

2.1.2.3 Overview of Early Adaptive User Interfaces in AR

Adaptive User Interfaces (AUI) often use machine-learning techniques to improve interaction. This enables the formation of an interface tailored to the abilities, disabilities, needs and preferences of the individual user. The major goals that any AUI should achieve are to assist the user to reach their goal more quickly, more easily, or to a higher level of satisfaction. Furthermore, AUI that aim to adapt to individual users must use some type of user modeling techniques. A user model is an explicit representation of the properties of an individual user and so it can be used to reason about user needs or preferences, or predict user behavior.

Ross (2000) listed four major factors in user model design for AUI. The first factor is the type of modelling, who it is for, whether it is the canonical user or individual user. The second factor focuses on the source of modeling information, whether the model is constructed explicitly by the user or Abstracted by the system on the basis of the user's behavior. The third factor determines the time sensitivity of the model: is it short-term for highly specific information or longer-term for more general information. The final factor relates to the update methods: is it a static or dynamic model. An example of a basic type of AUI is based on a static model that supports a canonical user, which is usually embedded implicitly into the system by default as opposed to modeling the individual user, where explicit methods and dynamic update are required to describe user state. The next section describes AR specific adaptabilities of AUI.

Julier et al. (2003) described five major user interface techniques that should be considered when designing an AUI for AR, information filtering, occlusion representation, adaptation to registration error, adaptive label placement, and 3D multimodal interaction.

Information filtering prevents information overload to the user in a complex visualization scenario (see Figure 2.9). Occlusion representation is required to realistically blend virtual contents into the physical world correctly (see Figure 2.10). Adaptation to registration error is necessary to improve user satisfaction to alleviate the effect of time-varying registration error that affects the alignment of the UI and the world (see Figure 2.11). Adaptive label placement must handle a dynamic annotation of the scene to prevent cluttered or ambiguous UI labeling that may arise (see Figure 2.12). 3D multimodal interaction is also crucial for development of wearable and ubiquitous computing.

Figure 2.9: Information filtering between unfiltered (left) and filtered display (right) (Julier et al., 2003)

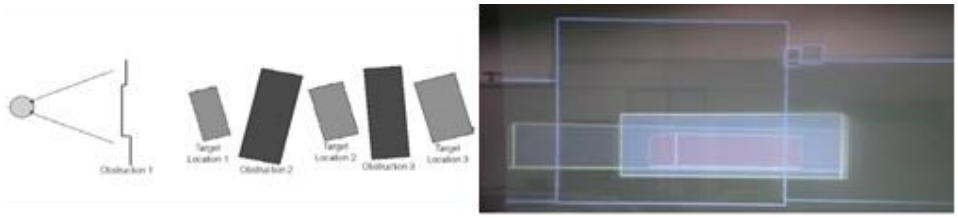


Figure 2.10: Occlusion representation (Julier et al., 2003)

Figure 2.11: Correction of registration error where the first case the windows are sufficiently far apart so they can be drawn unambiguously (left) but in the second case an aggregated window must be used instead (right) (Julier et al., 2003)

Figure 2.12: Unmanaged labeling (left) and managed labeling (right) (Julier et al., 2003)

Perritaz et al. (2009) studied the user experience in AR for the deployment of real-time adaptation. They determined the frame rate, image size, head motion speed and end-to-end delay were crucial variables that impact the user experience. They proposed a model to link the effect of the key variables with the Quality of Experience metrics. With their model, they proposed an adaptation scheme that would adjust, in real time, the frame rate and image size to improve the Quality of Experience while conforming to the rate constraint imposed. Their simulation showed that adaptation resulted in a better Quality of Experience and also out performed a solution with a fixed frame rate set to its maximum.

2.1.2.4 Overview of Early Multimodal User Interface in AR

Bolt (1980) created the first multimodal user interface (MUI) with the "Put-That-There" application. Going beyond the GUI, keyboard and mouse, "Put-That-There" processed spoken commands linked to a pointing gesture. Since then, researchers have continually improved the hardware and software components for MUI and explored new combinations of modalities such as gesture, pen, gaze tracking and lip movement, all combined with speech as the primary modality.

Oviatt (S. Oviatt, 1999) summarized empirical findings regarding the engineering of the MUI in "Ten myths of multimodal interaction" that raised important research questions such as "Do users interact multimodally providing a system with multimodal interaction?", "Is speech and pointing the dominant multimodal integration pattern?". Later, Oviatt et al. (S. L. Oviatt et al., 2000) compared the differences between GUI and MUI. They pointed out that the characteristics of GUI are having single input, being atomic and deterministic, using sequential processing, and having a centralized architecture, whereas, MUI supports multiple input streams, the input is continuous and probabilistic,

using parallel processing, and having a distributed and time sensitive architecture.

Reeves et al. (2004) indicated two primary objectives in MUI design. Firstly, it is desirable to achieve an interaction closer to natural human-human communication and, secondly the robustness of the interaction can be increased by using redundant or complementary information. Dumas, et al. (2009) conducted a comprehensive survey underlying the principle model of MUI where the model of multimodal man-machine communication was developed as shown in Figure 2.13 Both man and machine possess a similar communication pattern that can be divided into four states. The human makes a decision about the means of communication, then takes action, and in turn perceives the return messages and interprets them. The machine receives the input data and interprets those instructions and carries out appropriate computation, in turn it presents the information back to the user.

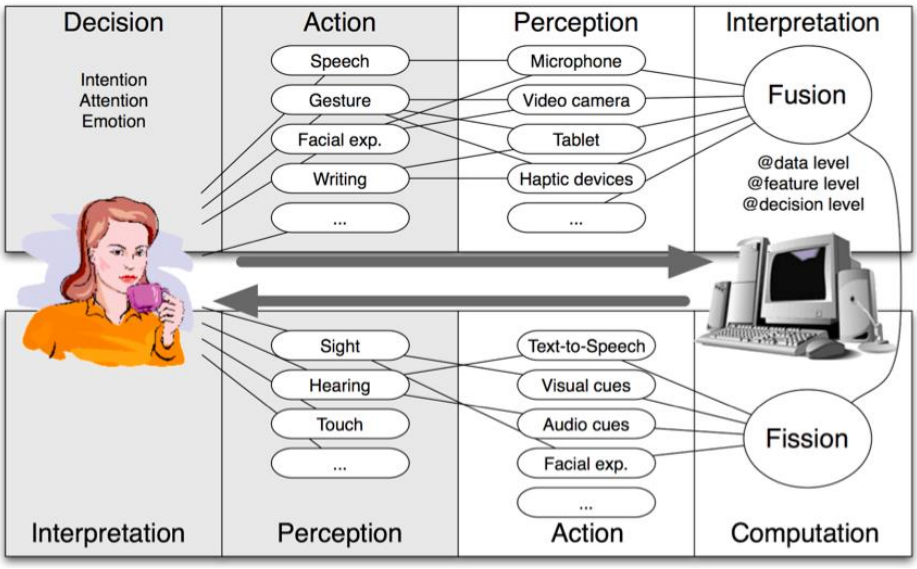


Figure 2.13: A representation of multimodal man machine interaction loop (Dumas et al., 2009)

Generally, there are two formal models that can be considered for MUI modeling. The first model is the CASE model (Nigay & Coutaz, 1993) that is based on four properties; concurrent, alternate, synergistic and exclusive. Each property represents a different approach for integration at the fusion engine, between combined or independent and sequential or parallel, as shown in Table 2.1. The second model is the CARE model (Coutaz et al., 1995). It also applies four properties; complementarity, assignment, redundancy and equivalence, where complementarity means modalities that complement each other, assignment means only one modality can yield the desired result, redundancy indicates modalities can be used together or individually to achieve the goal and equivalence means one modality can achieve the same result as another but only one can be used at a time.

Table 2.1: CASE model (Nigay & Coutaz, 1993)

		Use of Modalities	
		Sequential	*Parallel*
Fusion of Modalities	*Combined*	Alternative	Synergistic
	Independent	Exclusive	Concurrent

Multimodal fusion is a crucial feature of MUI that is required for interpreting the input from across modalities. Fusion can be executed at three levels (Dumas et al., 2009) (Lalanne et al., 2009); (1) *data level fusion* integrates multiple signals from similar source, (2) *feature level fusion* integrates tightly coupled or time synchronized modalities, and (3) *decision level fusion* integrates loosely coupled modalities. Table 2.2 shows the characteristics of fusion at each level.

Decision-level fusion is the most common fusion method since it can manage loosely coupled modalities. There are three common architectures for the decision-level fusion; they are (1) *Frame-based fusion* which uses frames or features to represent input data, (2) *Unification-based fusion* which merges attribute-value structures to obtain a logical whole representation, and (3) *Symbolic/statistical fusion* which combines statistical processing techniques with the above fusion techniques. Decision-level fusion has the advantage of mutual disambiguation that involves disambiguation of signal or semantic-level information in one error-prone input mode from partial information supplied by another input mode. This leads to error suppression within multimodal architectures.

Table 2.2: Characteristics of fusion levels (Dumas, Lalanne, & Oviatt, 2009)

	Data-level fusion	Features-level fusion	Decision-level fusion
Input type	Raw data of same type	Closely coupled modalities	Loosely coupled modalities
Level of information	Highest level of information detail	Moderate level of information detail	Mutual disambiguation by combining data from modes
Noise/failures sensitivity	Highly susceptible to noise or failures	Less sensitive to noise or failures	Highly resistant to noise or failures
Usage	Not really used for combining modalities	Used for fusion of particular modes	Most widely used type of fusion
Application examples	Fusion of two video streams	Speech recognition from voice and lips	Pen/speech interaction

In general, MUI has been thoroughly researched. Even though they were specifically designed in the context of AR, many of the findings are applicable in AR. For example Kaiser et al. (2003) created a hand gesture-speech interface

to manipulate virtual objects in an AR environment. They used a unification-based fusion to match the highest scoring interpretation from each input recognizer to produce a command. In detail, the speech and gesture signals are recognized in parallel, and both are time-stamped. The speech recognizer produces the n-best list of output strings and the gesture recognizer handles the 3D coordinate and finds the object corresponding to the pointing direction, which also compiles into n-best list. Finally, the probabilities score for speech gesture and object recognition are multiplied and the top-scored combination command is executed.

Heidemann et al. (2004) presented a hand gesture-speech interface for object selection and identification using machine learning for object recognition. However, they only gave a short description of their method of integration between modalities. They described each input module as being independent and provided a continuous stream of processing results. The control module processed the result as a state machine and so the output depended on the current state.

Kolsch et al. (2006) introduced a system for virtual objects manipulation using four input modalities; hand gesture, speech, unidirectional trackball motion, and head orientation. The system used unification-based fusion where recognizers process each input channel independently. However, the interpretation modes differed for different commands and system state. There are three modes: (1) *Independent and concurrent interpretation* takes atomic commands such as speech commands, (2) *Singular interpretation of redundant commands* takes commands from one channel that can be substituted by commands from another, and (3) *Sequential mode interpretation* takes commands that required complementary inputs from one channel and then from another consecutively.

Lee and Billinghurst (2008) supported hand gesture and speech input and the MUI had been implemented using statistical fusion. From the study, guidelines for designing MUI for AR were: (1) use phrase-based speech commands, (2) use a fast gesture recognition module, (3) use gesture triggered multimodal fusion, (4) use audiovisual feedback, and (5) use learning modules in the multimodal fusion architecture.

2.2 Augmented Reality Interaction Techniques Prior to Depth Sensing Era

In the early 1990s, AR interfaces emphasized intuitive methods for viewing three-dimensional information in application domains such as medicine and machine maintenance (Feiner et al., 1993). Later researchers attempted to move further than visualization and provided support for content creation and manipulation in AR. For example Kiyokawa et al. (1999) provided users with a 3D user interface which allowed them to create AR content. Schmalstieg et al. (2000) and Butz et al. (1999) used tracked pens and tablets for AR objects selections and modification.

In 1997, Ishii and Ulmer (1997) introduced the idea of Tangible User Interfaces (TUIs), where users can interact with digital information in real space through physical objects (Ishii, 2008). Billinghurst et al. (2008) extended this to create the Tangible Augmented Reality (Tangible AR) interface metaphor which yielded a number of new interaction techniques (Billinghurst et al., 2009; Lee et al., 2011; Looser et al., 2007). Furthermore, natural interaction techniques such as gesture (Buchmann et al., 2004; Fernandes & Fernandez, 2009; Lee et al., 2008), speech, haptic (Guangqi et al., 2003; Rhienmora et al., 2010), gaze and multimodal (Harders et al., 2007; Heidemann et al., 2004; Hurst & Van Wezel, 2011; Irawati et al., 2007; Kaiser et al., 2003; Kolsch et al., 2006; Olwal

et al., 2003), are active fields of research into uncovering the potential of AR interfaces and their applications.

2.2.1 Emergence of Interaction Techniques in Augmented Reality

Billinghurst et al. (2009) proposed four stages in the development process that a new interface technology often passes through: (1) prototype demonstration, (2) adoption of interaction techniques from other interface metaphors, (3) development of new interaction metaphors appropriate to the medium, and (4) Development of formal theoretical models for user interaction. Today, researchers are exploring old and new paradigms as AR technology is maturing. Surveys into AR research provided insights into the AR development based on existing metaphors from desktop or VR environments (van Krevelen & Poelman, 2010; Zhou et al., 2008), and recently researchers have started exploring new paradigms of user interface suitable for AR. In this section we present research into interaction techniques in AR prior to depth sensing era, with tangible augmented reality interface and interaction in Section 2.2.1.1, hand gesture interaction in Section 2.2.1.2, haptic interaction in Section 2.2.1.3, and various combination of multimodal interaction in Section 2.2.1.4 to 2.2.1.8.

2.2.1.1 Tangible Augmented Reality Interface and Interaction

Borrowing the concept of Tangible User Interfaces (TUI) developed by Ishii and Ullmer (1997), Billinghurst et al. (2001) utilized physical objects as input devices. The idea matured through studies of collaborative scenarios, including a project called Shared Space (Billinghurst et al., 2000). These Tangible AR interfaces attach a virtual object to a physical one such that users can interact with virtual objects by directly manipulating the tangible objects. They strongly believed that by combining TUI and AR, an intuitive and seamless method of

interaction and display could be achieved. They summarized the design principles from this interface as: (1) use physical controllers for manipulating virtual content, (2) enable spatial 3D interaction techniques (such as using object proximity), (3) Support for both time-multiplexed and space-multiplexed interaction, (4) enable multi-handed interaction, (5) match the physical constraints of the object to the task requirements, (6) permit parallel activity where multiple objects are being manipulated, (7) collaboration between multiple participants.

Tangible AR (Billinghurst et al., 2005) has been around for over a decade and there has been a number of publications which have contributed to the area. The rest of this section presents the major works, all of which use the ARToolKit (2015) library for tracking.

Kato et al. (2000) created a paddle with a marker attached to the end to prevent occlusion from the hand. The combination of a paddle and an array of markers on the tabletop could be used to manipulate virtual objects, as they illustrated through an interior design application. Two types of gestures could be made with the paddle; static and dynamic. Static gestures were based on the proximity to object and paddle tilt and inclination. Dynamic gestures support actions, including a side-to-side movement imitating shaking, and up and down movement imitating hitting and pushing.

Looser et al. (2007) created a novel visualization technique called the Magic Lens. They introduced a flexible Tangible AR interface that supported multiple poses, allowing manipulations of the lens such as stretching, twisting, bending and fanning. Programmable physical buttons were integrated into the handle of the tangible interface, see Figure 2.14 (left).

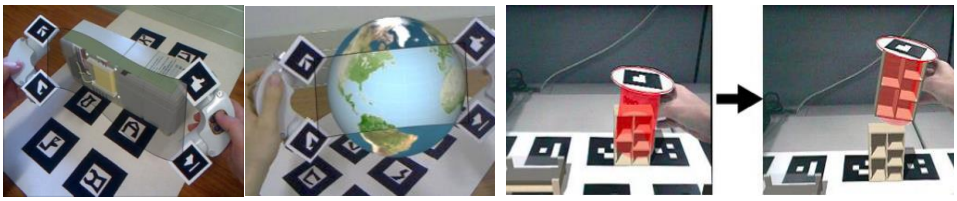


Figure 2.14: Magic Lens (Looser et al., 2007) (left) and MagicCup (Billinghurst et al., 2009) (right).

Billinghurst et al. (2009) demonstrated an intuitive cup-shaped tangible AR interface called the MagicCup as shown in Figure 2.14 (right). The MagicCup acted as a placeholder for the virtual object and allowed the user to perform actions on the object such as pick, move, rotate, delete etc.

Recently, Lee et al. (2011) created a cube-based tangible AR interface, called the cubical user interface (CUI), which allowed a user to compose objects using both of their hands to manipulate the blocks (See Figure 2.15). The block has markers on each of its faces, resulting in robust tracking and free rotation. By holding the block in each hand, the objects can be composed by imitating a screw-driving action between blocks. Furthermore, the block also has physical buttons on every corner, which are programmable and communicate wirelessly through Bluetooth.

Figure 2.15: CUI makes two hands tangible AR possible (H. Lee, Billinghurst, & Woo, 2011)

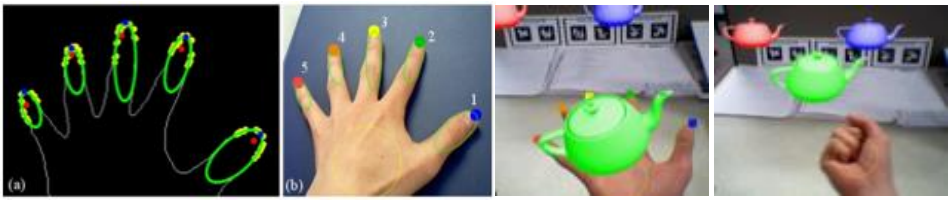


Figure 2.16: Handy AR (T. Lee & Hollerer, 2007), fingertips detection (left) and object selection (right)

2.2.1.2 Hand Gesture Interaction

Lee and Hollerer (2007) created Handy AR, which used a bare-hand instead of a marker for tracking using a standard camera. By using skin color segmentation and taking the largest blob as the hand, fingertips could be found by fitting ellipses to contours based on the candidate points. Furthermore, by using ARTag to stabilize virtual objects in the scene, the hand could be used for object selection and inspection. However, for the detection to work well, users must stretch out their hands as shown in Figure 2.16. The supported gestures were open/close palm for an object selection and turn around/over the hand for an inspection.

Later, they extended their work to allow markerless tracking (Lee & Hollerer, 2009). By computing optical flow of features points, a world coordinate could be established using hand tracking from the previous work as a reference to the environment. Once the coordinate system was established, the tracked scene could be extended and the tracked hand could be used for interaction with virtual objects such as moving and placing objects onto the tracked scene as illustrated in Figure 2.17.

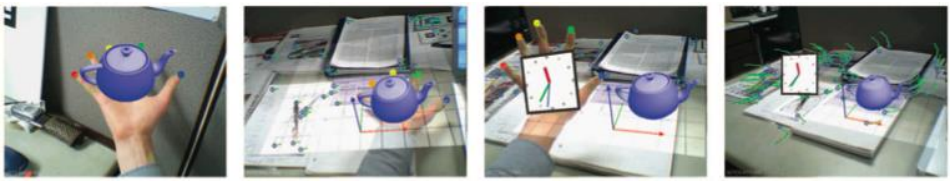


Figure 2.17: Handy AR and markerless tracking on desktop workspace (Lee & Hollerer, 2009)

Fernandes and Fernandez (Fernandes & Fernandez, 2009) used hand images for training statistical models to detect the hand. The Haar-like features were combined with the AdaBoost algorithm to extract the features characteristics of the hands. By using ARTag for tracking, virtual objects could be moved using a palm up posture. For rotation and resizing, both hands were required as shown in Figure 2.18.

Figure 2.18: Hand gestures over ARTag (Fernandes & Fernandez, 2009)

Hurst and van Wezel (2011) created a multimodal interface on mobile AR platform. Their interface provided three types of interaction; (1) touch screen based, (2) position and orientation of the device based using the integrated accelerometer and compass, and (3) finger based where a small marker was put on the finger and tracked by the device camera. There were three tasks in the study; object selection, menu selection and translation. The touch screen based interaction achieved the shortest time for both selection tasks and was rated the highest for performance. The device input had the shortest time for the translation task and was highest ranked for this task. The finger interaction performed poorly in term of performance but users found it to be engaging and fun, indicating potential for games and leisure applications on mobile devices.

Datcu and Lukosch (2013) used free-hand gestures to control a menu system on an HMD with a stereo camera. Their system supported 6 *dof* hand tracking and hand pose recognition through a vision-based algorithm (Akman et al., 2013). Hand poses were mapped to commands for menu selection. Later, they conducted a study comparing free-hand input and ordinary physical objects for

operating the menu-based interface (Datcu et al., 2015). They found using physical objects to be beneficial as it provided tactile feedback.

2.2.1.3 Haptic Interaction

Harders et al. (2007) presented a haptic AR application for medical training scenarios such as intra-operative surgical navigation. They used optical tracking for tracking markers and head position. Interaction was achieved through direct manipulation of the haptic device with the virtual object on the marker, illustrated in Figure 2.19 (left). With calibration of the world and haptic device coordinate, the real and virtual objects could be interacted with seamlessly in the same environment.

Rhienmora, et al. (2010) created a dental surgical skill training utilizing a haptic interaction. By using ARToolKit for tracking, world and haptic coordinates were co-located, allowing direct dental operation on the virtual tooth as shown in Figure 2.19 (right).

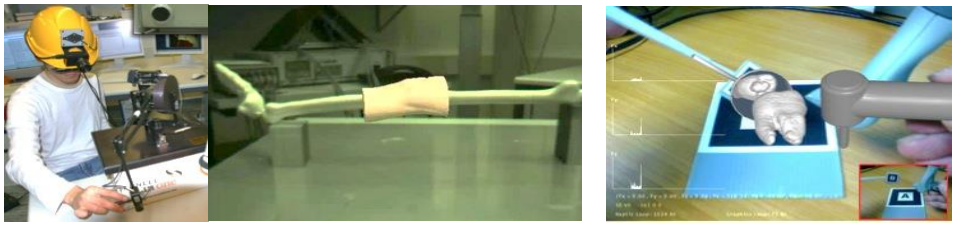


Figure 2.19: Haptic AR application for medical training (Harders etal., 2007) (left) and dental surgical skill training (Rhienmora et al., 2010) (right)

2.2.1.4 Multimodal Interaction: Hand Gesture and Speech

Kaiser et al. (2003) integrated 3D gesture, gaze and speech into MUI for AR and VR environments. Integrating spatial correlation between deictic terms, such as “that”, “here”, and there”, in an object selection task, they created a system called SenseShapes (Olwal et al., 2003). The system calculated

statistically the region of interest from the given speech, gaze projection from head tracking and pointing projection through glove-based finger tracking. An object that lay within the intersection was stored in the database at each frame. It could project four types of volume primitive: cuboids, cylinders, cones, and spheres. Figure 2.20 (left) shows SenseShapes projecting a green cone from the finger pointing that intersects with a black chair on the left and the data glove for tracking fingers on the right.

Heidemann et al. (2004) demonstrated an AR interface that could identify objects on the tabletop as shown in Figure 2.20 (right). By using a VPL-classifier, the user could train the system to identify new objects. The skin color segmented index finger could be used for pointing at objects and making menu selections. Speech could be used for querying information and interacting with the menu as well. The system provided visual feedback with 2D graphics with such as boxes and labels on the objects.

Figure 2.20: MUI for AR (Kaiser et al., 2003) (left) and Heidemann et al. (Heidemann, Bax, & Bekel, 2004) (right)

Kolsch et al. (2006) created a mobile AR system that supported hand gesture, speech, trackball and head pose as shown in Figure 2.21. Gesture was implemented using HandVu (2015), a computer vision module that allowed hand-gesture recognition. They categorized tasks by the dimensionality required. Taking a snapshot was considered 0D, adjusting the focus region depth was 1D, using a pencil tool for finger was 2D, while orienting virtual objects was 3D. Actions such as take/save/discard snapshot could only be performed by

a single modality that was speech, but other actions such as relocate/resize/orient could be performed multimodally by speech, gesture or trackball. Their system included a tunnel tool (supporting slicing and x-ray vision) for visualization, virtual object manipulation, and a path finding and navigational guidance capability.

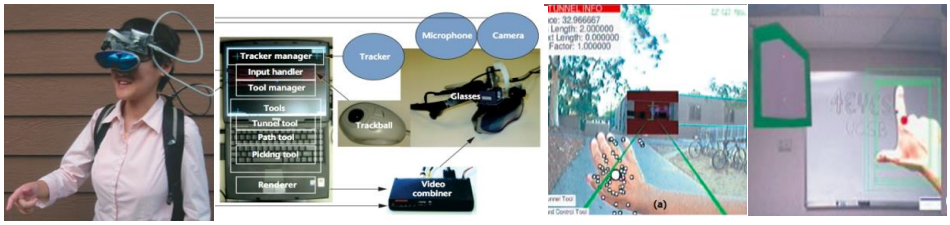


Figure 2.21: Mobile AR with MUI utilizing HandVu for hand gestures (Mathias et al., 2006)

Lee (2010) also used skin color segmentation to allow bare hand gesture with speech input as shown in Figure 2.22 (left). They carried out user-centered experiments and improved interaction techniques using methods such as providing hand occlusion of virtual objects. They conducted the first Wizard of Oz (WOz) study for gesture and speech input in AR to acquire interaction patterns from users. Later, they conducted a usability study of the hand gesture-speech multimodal interface comparing between speech only, gesture only and multimodal. There were ten tasks to complete with virtual objects using three commands; (1) change color, (2) change shape, and (3) move the object. It was found that gesture input was considered more natural than speech alone or the combination of both. However, multimodal interface usage and the time window for combining gesture and speech were found to be dependent on the type of task and contributed to 68% of input. Overall, the multimodal interface produced shorter task completion time and used less commands but produced a higher number of error. Despite the lower effectiveness, users preferred the multimodal interface.

A more recent study compared speech-only, gesture-only, and multimodal input in AR for translation and changing shape and color tasks (Minkyung Lee, Billinghurst, Baek, Green, & Woo, 2013). They found that the multimodal condition was more usable than the gesture-only interface, and was more satisfying to use than the speech-only condition. However, the study was only conducted for tasks that involved translation on a 2D planar surface.

2.2.1.5 Multimodal Interaction: Hand Gesture and Haptic

Guangqi et al. (2003) introduced an AR system called VisHap, a framework using visual tracking to integrate force feedback with tactile feedback to generate a haptic experience. The three components of this system were the vision subsystem, haptic subsystem and AR environment subsystem. In the vision subsystem, a stereoscopic camera was used to capture image pairs for calculating the disparity and the 3D data. Skin color segmentation was used to identify the user's fingertip position. The combined system allowed for haptic and finger-based interaction.

Buchmann et al. (Buchmann et al., 2004) used markers to track the thumb, index finger and the pivotal point on the hand, allowing the user to directly manipulate virtual objects with the hand. The system was called FingARtips as illustrated in Figure 2.22 (right).r Furthermore, a buzzer was attached to the tip of the finger to provide haptic feedback when the finger was in contact with an object.

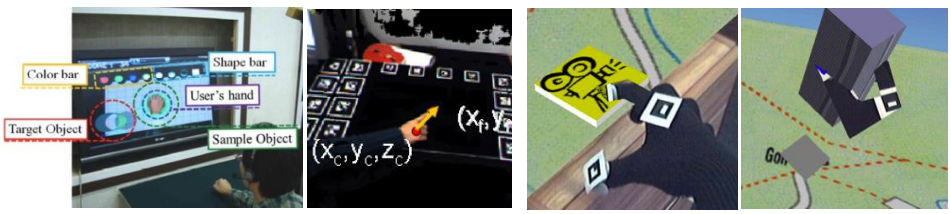


Figure 2.22: AR MMI with skin-color hand segmentation (Lee, 2010) (left) FingARtips with tactile feedback (Buchmann et al., 2004) (right)

2.2.1.6 Multimodal Interaction: Tangible and speech

Irawati et al. (Irawati et al., 2006, 2007) combined paddle gestures and speech to manipulate objects in AR (see Figure 2.23). For display, a video see-through HMD was used with a camera attached in front. In their interior design application, speech commands supported included "Select a desk", "Place here" and "Move the couch". They conducted a user study to compare three conditions; paddle gestures only, speech with static paddle position, and speech with paddle gestures to build three different furniture configurations. They found that a combination of speech and paddle gestures improved the efficiency of user interaction

Figure 2.23: MUI utilizing VOMAR, a paddle-based Tangible AR (Irawati et al., 2007)

2.2.1.7 Multimodal Interaction: Tangible and Haptic

Aleotti et al. (2010) demonstrated haptic AR with physics-based animation that supported both rigid and deformable bodies (see Figure 2.24). The AR

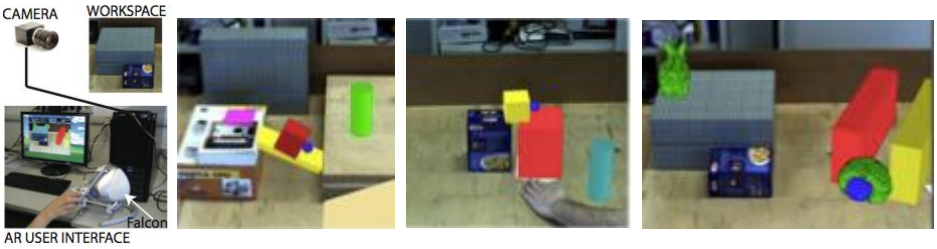


Figure 2.24: Another haptic application that permits collaboration with tangible interface (Aleotti, Denaro, & Caselli, 2010)

workspace contained registered static physical objects representing the tangible interface, virtual objects, and proxy sphere that represented the 3-degree of freedom haptic interface position in AR space. They found that force feedback increased the efficiency and reduced user task completion time. A collaborative scenario was also carried out, where one user used the haptic device and another user could move the fiducial marker to interact with the virtual objects. However, their workspace was static which meant that new physical objects could not be introduced into the workspace in real-time, unlike other tangible interfaces.

2.2.1.8 Multimodal Interaction: Hand Gesture, Tangible and Haptic

Lee et al. (2010) used glove-based interaction for both gestures and tangible input, which aimed at indirect and direct manipulation, respectively. In their work, they referred to two types of interaction techniques, soft interaction (such as hand gesture) and hard interaction (such as vibro-tactile feedback). The AR environment and glove were tracked with ARToolKit markers, with the fiducial markers placed around the wrist area of the glove for tracking hand position and rotation (see Figure 2.25). The glove was specially made with conductive fabric on the fingertips and the palm for gesture recognition. It also contained vibration

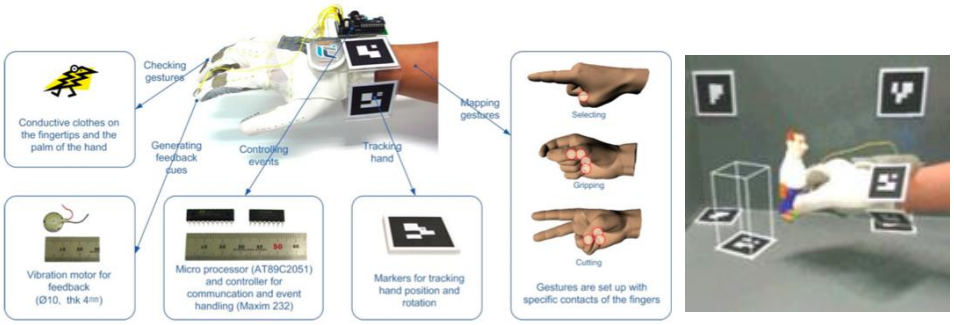


Figure 2.25: Vibro-tactile data glove allows direct manipulation of virtual objects(J. Y. Lee, Rhee, & Seo, 2010)

motors for haptic feedback when fingers touched the virtual objects. Actions supported by gesture included selecting, gripping, cutting and copying.

2.2.2 Overview of Environment Awareness and Physically-based Interaction

2.2.2.1 Environment Awareness in AR

We define environment awareness as real-time awareness of physical changes in the environment that the system is monitoring. This permits interaction between physical and virtual objects through physical simulation. Awareness of the environment within AR is a well-researched area. It is required to achieve correct occlusion (Lepetit & Berger, 2000), collision detection (Breen etal, 1995), and realistic illumination and shadowing effects (Wang & Samaras, 2003). While these features are not necessary for Augmented Reality, it has been shown that applications which include such cues can establish a stronger connection between real and virtual content (Sugano et al., 2003).

Early attempts at environment awareness required manual modeling of all the real objects in the environment, and online localization of the camera to ensure virtual objects interact with real objects appropriately (Breen et al., 1995; MacIntyre et al., 2005). This method is both time consuming and inflexible, as any changes in the environment will require recalibration.

Later approaches involved more automatic techniques, such as contour based object segmentation (Berger, 1997), depth information from stereo cameras (Zhu et al., 2010) and time-of-flight cameras (Fischer et al., 2007), or online SLAM (Ventura & Hollerer, 2009). By automatically acquiring the relevant information from the scene, there is no offline calibration requirement, and the system can correctly process the environment even when objects change or are added and removed. However, these techniques often fail in untextured

scenes due to homogeneous objects, poor illumination or require an initialization process.

2.2.2.2 Physically-based Interaction in AR

Physical simulation in AR can be achieved through the integration of a physics engine into the AR framework. A physics engine is a software component that can simulate physical systems such as rigid body and soft body dynamics. There are both open source physics engines such as the Bullet physics library (Bullet Physics Engine, 2015), Newton Game Dynamics (Newton Dynamics, 2015), Open Dynamics Engine (ODE) (Open Dynamic Engine, 2015), and Tokamak physics engine (Tokamak Physics, 2015), and proprietary engines such as Havok (Havok Physics, 2015) and Nvidia's PhysX (Nvidia PhysX, 2015).

The use of physical simulation in AR has been thoroughly researched. Song et al. (Song et al., 2008) developed two applications, Finger Fishing and Jenga using Newton Game Dynamics. Optical Stereo based Fingertip detection was used to find the tip of an index finger, allowing simple interaction such as picking and dragging (Figure 2.26a). However, without the use of a reference marker, the camera was not movable and could only provide a fixed and pre-calibrated view.

Buchanan et al. (Buchanan et al., 2008) integrated ODE and OpenSceneGraph (OpenSceneGraph, 2015) to simulate and render rigid body dynamics in their educational application about abstraction of forces (Figure 2.26b). MacNamee et al. (MacNamee et al., 2010) created a tabletop racing game and a forklift robot simulation using ODE and Havok, respectively, and used OpenGL (OpenGL, 2015) for graphics rendering (Figure 2.26c). Both used ARToolKit (ARToolKit, 2015) library for tracking fiducial markers. In these applications, the physical interaction was limited to a pre-defined physics proxy that was represented by a corresponding marker.

Wilson demonstrated using depth-sensing camera to reconstruct the surface of the interactive area of the table and create terrain for a car racing game, Micromotocross, as shown in Figure 2.26d (Wilson, 2007). It was arguably one of the most advanced uses of depth sensor at the time, however, the display used was two-dimensional projection onto the tabletop with no support for a 3D display such as AR.

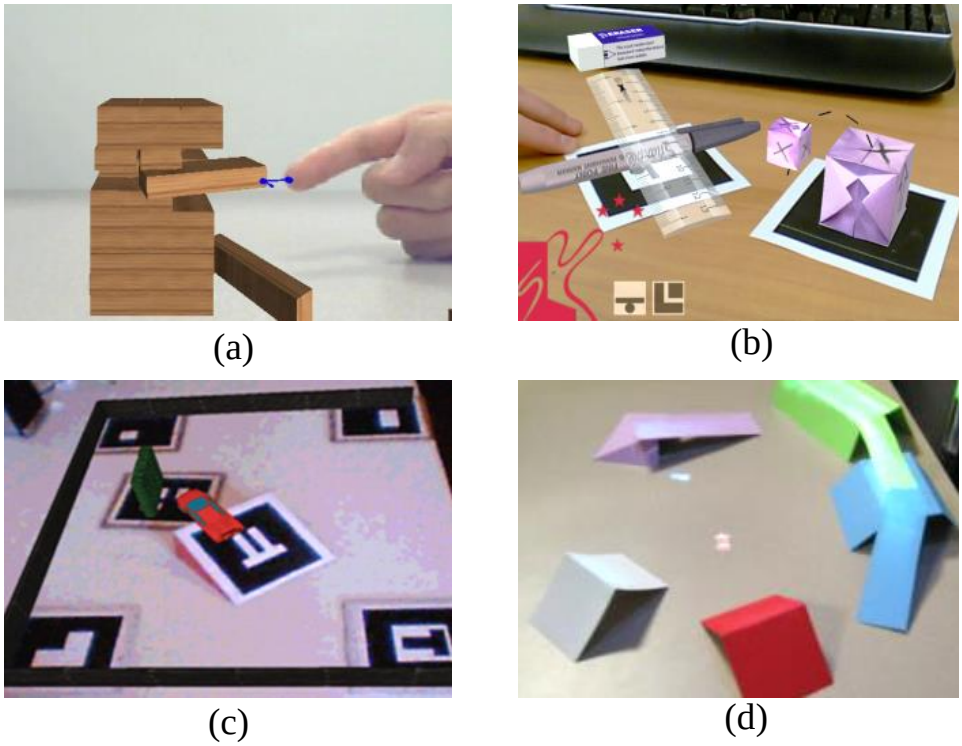


Figure 2.26: (a) Physically-based fingertip interaction in Jenga (Song et al., 2008) (b) Rube Goldberg machine in AR (Buchanan et al., 2008) (c) A car accelerates up the fiducial marker slope in a car racing AR (MacNamee et al., 2010) (d) Two virtual cars are projected on the tabletop racing up the real tangible ramps in Micromotocross (Wilson, 2007).

2.3 Overview of Depth Sensing Technology and its Impact on Natural Hand Tracking and Interaction

One of the key technologies that has created a recent paradigm shift in Human Computer Interaction (HCI) is depth sensing and algorithms that have enabled vision based tracking of human body. The Microsoft Kinect (Microsoft Kinect SDK, 2015), a low cost consumer depth sensors released in late 2010, has been used in many fields and applications from healthcare to robotics (Zhang, 2012). The widespread use of this technology has significantly advanced research and development in HCI over the past few years. In this section, we present existing research that has applied depth sensing technology to supporting AR interactions. First, we give an overview of depth cameras and natural interaction platforms that have been used in recent AR systems in Section 2.3.1. Then we briefly describe research in hand pose estimation and recognition in Section 2.3.2. Research that utilizes depth sensors for natural interaction is covered in Section 2.3.3.

2.3.1 Overview of the Depth Sensors and Natural Interaction Platform

In this section, we give examples of low cost depth cameras and natural interaction platforms that support depth input in Section 2.3.1.1 and 2.3.1.2, respectively.

2.3.1.1 Consumer Depth Cameras

The two types of consumer depth cameras that have been widely used in AR interface and interaction in recent years are (1) structured light cameras, and (2) Time-of-flight cameras.

2.3.1.1.1 STRUCTURED LIGHT CAMERAS

Structured light cameras illuminate the scene with a structured light pattern so that the depth at each pixel can be determined from a single image of the reflected light (Batlle et al., 1998). Low cost depth cameras that use this technology include the first generation Microsoft Kinect (Microsoft Kinect SDK, 2015), and cameras based on the Primesense depth sensor (Primesense, 2015) (See Figure 2.27), such as the Primesense Carmine and Asus Xtion. These cameras project infrared light onto the scene with a speckle pattern and use depth from focus and stereo computer vision techniques to generate a depthmap of the scene.

Figure 2.27: Microsoft Kinect (Microsoft Kinect, 2015) (left) Primesense depth camera (Primesense, 2015) (right)

2.3.1.1.2 TIME-OF-FLIGHT CAMERAS

Based on the speed of light, the time-of-flight (*ToF)* camera, measures the time taken for a light signal leaving the camera to reflect back from the subject in the scene to determine the distance for each point on the image. The low cost solutions that are available include the second generation Microsoft Kinect (Microsoft Kinect SDK, 2015), Structure sensor (Structure Sensor, 2015), Intel RealSense (Intel RealSense, 2015), SoftKinetic (SoftKinetic, 2015), and PMD Camboard (PMD Technologies, 2015) (See Figure 2.28).

Figure 2.28: Second generation Microsoft Kinect (Microsoft Kinect, 2015) (left), Structure sensor (Structure Sensor, 2015) (middle left), Intel Realsense (Intel Realsense, 2015) (middle right), and PMD Camboard (PMD

2.3.1.2 Natural Interaction Platforms based on Depth Sensors

Arguably the most popular platform to date, the Microsoft Kinect SDK (Microsoft Kinect SDK) is a natural interaction platform for the Windows operating system that supports full body tracking, face tracking, and speech recognition. Another popular platform is the Intel RealSense SDK (Intel RealSense), originally called the Intel Perceptual Computing SDK, which supports natural interaction including hand and finger tracking, face tracking, speech recognition and synthesis, and runs on the Windows and Android operating systems. Prior to their acquisition by Apple, Primesense launched an Open Source platform for natural interaction called OpenNI (Primesense), which offers full body tracking. SoftKinetic offers a platform that supports full body, hand and finger tracking through the Iisu middleware (SoftKinetic). The Nimble SDK offers high degree-of-freedom (DOF) hand and finger tracking, and was integrated with the Oculus Rift to create NimbleVR (NimbleVR, 2014), which provides natural hand input for head-mounted displays (HMD).

2.3.2 Hand Pose Estimation and Recognition with Depth Sensing

Natural hand tracking and interaction has been the focus of many research areas, especially in computer vision and human-computer interaction. One benefit from such input is that users do not need to be encumbered with external peripherals such as sensing gloves. The two main advances in technology that

have allowed for new approaches for hand pose estimation and recognition are the release of consumer depth sensors, which provide accurate real-time depth information and parallel computing on the GPU, especially through the Nvidia CUDA API (Nickolls et al., 2008) that permits many algorithms to be parallelized and executed in real-time on a personal computer. The affordability of both depth sensors and graphics cards as well as the advancement in the corresponding application programming interface (API) for those platforms, has made them more accessible to both researchers and end consumers.

Erol et al. (2007) compiled a comprehensive survey of vision-based hand pose estimation research, showing that there were two main approaches. The first approach was partial pose estimation where only specific parts of the hand were being tracked, such as the tip of the finger. This reliance on appearance-specific image analysis only enabled a low degree of freedom (DOF) approximation of the pose. The second approach was the full DOF estimation where the hand position, orientation and joint angles were being estimated.

There are two approaches to a full DOF approach. The first is model-based tracking that represents the hand with a parameterized 3D model where an algorithm searches for parameters that match the features from the observation using the initial observations and dynamics to make the prediction. The second approach is single frame pose estimation, which operates independently of temporal information and is robust to rapid hand movement compared to model-based tracking that relies on an initial pose estimation.

One state of the art solution to model-based hand tracking is demonstrated by Oikonomidis et al. (2012). They parallel GPU processing and a depth sensor to perform model-based pose estimation in real-time. Single hand (Oikonomidis et al., 2011) and later two hands (Oikonomidis et al., 2012) were tracked using Particle Swarm Optimization (PSO) that fitted a hand model to the actual hand observation. This method demonstrated robust and extremely high fidelity

tracking of single and multiple hands with self-occlusion. However, this method was still computationally expensive even on a GPU, with the tracking only running at 15 Hz, and as such this method may not be appropriate for an interactive system.

Another promising algorithm used a single frame depth image for hand pose estimation. Keskin et al. (2011) proposed this method based on the work of Shotton et al. (2011), who described a full body pose recognition using random forest to classify each body region using only the depth information. Keskin et al. adopted the same method but applied it specifically to the human hand. In this research, Keskin's approach has been adapted to train and predict each hand region, and can be executed in real-time. Further details of the library design and implementation are covered in the Chapter 5 of this thesis.

With the ability to calculate hand pose estimation using full *dof*, the resulting poses can be used for training gestures. Gestures can be static or dynamic, with the former referred to as postures and the latter as gestures. A recognizer for postures requires only spatial information of the current hand's state. However, a gesture recognizer requires spatio-temporal information where accumulative states must be considered. Mitra and Acharya (2007) present a comprehensive survey on gesture recognition and have summarized four approaches to solve the problem; (1) Hidden Markov Models (HMM), (2) Particle filtering and condensation algorithms, (3) Finite State Machines (FSM) , and (4) Soft computing and connectionist approaches.

2.3.3 Natural Hand Interaction with Depth Sensing

Based on past research, we classify natural hand interaction by the location where the interaction occurs: on-the-surface or in-the-air. In this section we review recent research categorized by this classification.

2.3.3.1 On-the-surface Interaction

Early research that used only a depth camera to detect touch on a tabletop was conducted by Wilson (2010). By mounting a depth camera looking downward onto any surface, flat or non-flat, multi-touch input can be detected (See Figure 2.29). A simple depth histogram of the background was created to determine the depth of the surface at each image pixel. A fixed depth threshold between two values above the surface depth was used to segment fingers. He found that the detection at the moment of touch with a depth sensor was not as precise as direct sensing techniques but there was a number of advantages such as the surfaces needing no instrumentation, permitting detection on non-flat surfaces and being able to use information about the non-touching states of user's hands and arms above the surface.

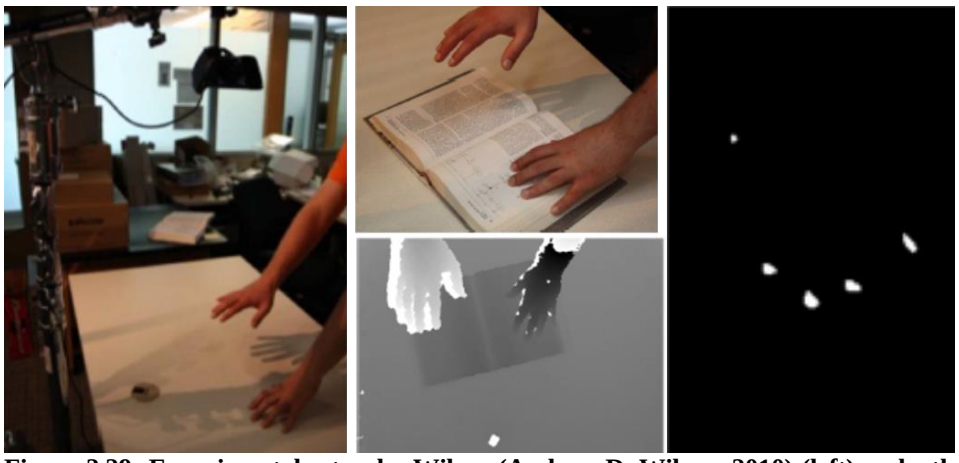


Figure 2.29: Experimental setup by Wilson (Andrew D. Wilson, 2010) (left), a depth image from the depth camera (middle-bottom), touch detected from segmented fingers (right)

In another work by Wilson et al (2008), they employed a physics engine to provide a natural interaction technique on a surface. Their interaction allowed the user to manipulate digital objects in a similar way as they would interact with real world objects. The system is able to model shape information and multiple contact points on the physical object, and can simulate friction and collisions. This permitted interaction using fingers, hands, and other physical objects. They introduced "proxy objects", a simulated rigid body created for each surface contact, which is kinematically controlled to interact with the other objects in the scene. This technique was extended to support particle proxies, where the contours of physical objects obtained from the Sobel image were represented by multiple particles (See Figure 2.30).

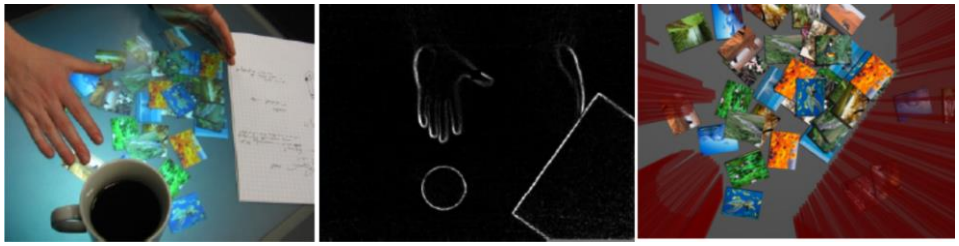


Figure 2.30: Physics-enabled on the surface (Andrew D. Wilson, Izadi, Hilliges, Garcia-Mendoza, & Kirk, 2008) (left), Sobel image showing the contours (middle), particle proxies represented in the physics simulation (right)

KinectFusion (Izadi et al., 2011) is a system that processes depth data from a moving Microsoft Kinect in real-time to track and reconstruct the environment. They showed that with the static model of the environment as a background, dynamic foreground objects such as a user's hand can be segmented and used to detect touch. This allowed for touch sensing on any surfaces that had been reconstructed by the system. A raw depthmap was taken as input and converted to vertex and normal map. The depth features were used to track the change in position of the camera, and the iterative closest point (ICP) algorithm was employed to compare the consecutive frames. The new data was

then integrated to voxel grids of the scene. Touch could be detected from changes on the background surface by separating the background and foreground scene (See Figure 2.31).

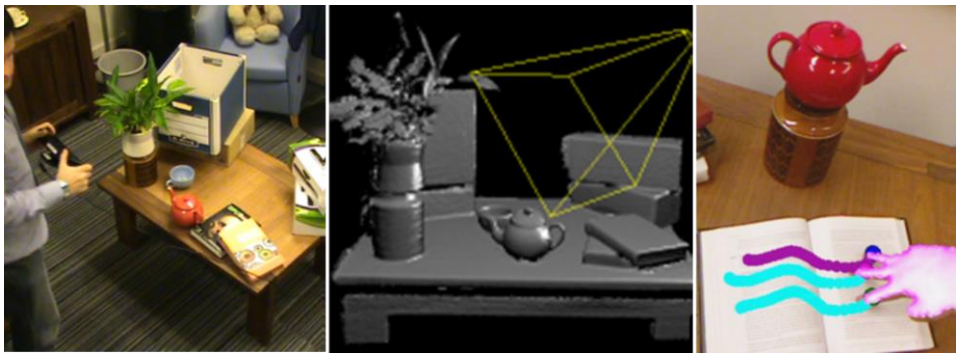


Figure 2.31: KinectFusion (Izadi et al., 2011); Microsoft Kinect is being moved around the scene (left), reconstructed scene and the current camera frustum (middle), multitouch on reconstructed surface (right)

2.3.3.2 Gesture In-the-air Interaction

Digits (Kim et al., 2012) is a wrist-worn camera-based sensor that estimated hand poses using infrared (IR) illumination. By solving the inverse kinematic (IK) from the estimated fingertips position, the finger joint position could be determined. In their experiment six hand postures were supported including open palm, showing index and middle fingers, pointing, grasping small object, grasping large object, and pinching. The usage scenarios included accepting phone calls and manipulating objects on large displays (See Figure 2.32).

Hilliges et al. (2009) extended the work of Wilson et al (Andrew D. Wilson et al., 2008) by offering interaction above a surface. The system supported pinch gestures that could be used to pick up a virtual object and move it above the computing surface. The finger positions were estimated and tracked using the contours extracted from the depth image. Using the depth of the tracked hand,

the object could be moved in the vertical direction above the surface (See Figure 2.33).

Figure 2.32: Digit is a wrist-worn sensor that recognizes hand gestures (Kim et al., 2012) (left), Digit can be used to interact with large display (middle), or answer a call (right)

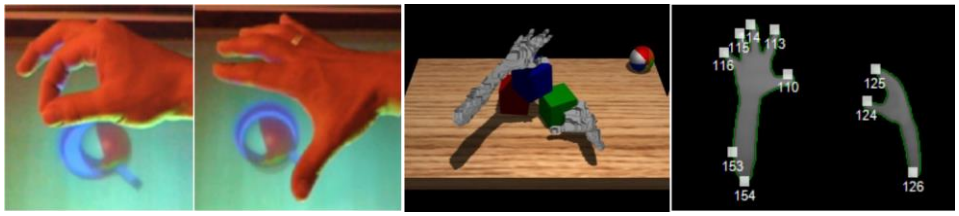


Figure 2.33: Interaction in the air, pinch and release an object (Hilliges et al., 2009) (left, middle left), simulation showing pointcloud of user hands (middle right), fingertip tracking (right)

2.4 State of the Art in Augmented Reality with Depth Sensing

This section discusses existing research in AR that employs depth sensing technologies. We categorize this research based on the type of display and affordance offered for different systems, with the following sections covering Spatial AR, transparent displays, and depth sensing in hand held devices and head mounted displays (HMD).

2.4.1 Spatial Augmented Reality (SAR)

Spatial Augmented Reality (SAR) overlays graphical information onto the real world using digital projectors. Since the display is independent from the user and the view can be shared among multiple users, SAR promotes collaboration in a collocated environment. The limitations of this display method are that privacy of the individual's data is not supported and the graphics overlay is limited to a 2D surface. In this section, prior SAR research has been divided by configuration into wearable, situated tabletop, and situated room-scaled.

2.4.1.1 Wearable SAR

Harrison et al. (2011) presented a depth sensing and projection system that users could wear on their shoulder that allowed multi-touch on ordinary surfaces, including parts of the user's body such as their palm or their arm. Finger segmentation was based an analysis of the depth derivative image using a sliding window to search for features that match that of a finger. Their user study results suggest that touch detection with depth sensing cameras is possible and can be very promising with improved resolution and precision of the sensor. Two interface placement methods, classification-driven placement and user-

specified placement, were proposed. Their system was limited due in distortion of the projection, it did not support posture and gesture input, and the projection and tracking space was fixed in-place (See Figure 2.34).

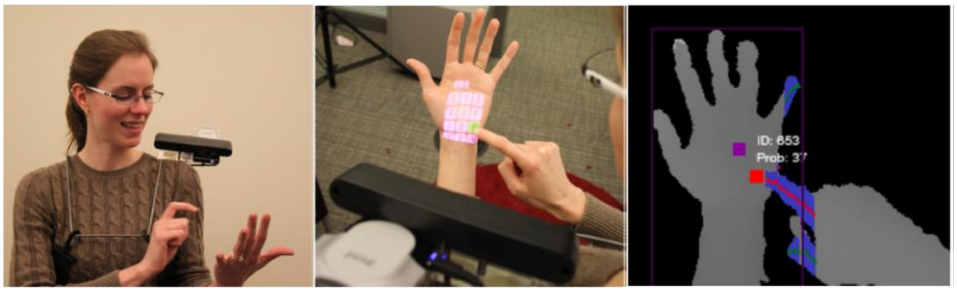


Figure 2.34: A projector and a depth sensor are mounted to user's shoulder (Harrison et al., 2011) (left), a user is using her hand as a touch interface (middle), finger segmentation and the estimated fingertip position (right)

2.4.1.2 Situated Tabletop SAR

In situated AR (Bimber & Raskar, 2005), the focus has primarily been on interaction in the tabletop arm-reach distance space. Wilson employed a depth camera to create a 3D tangible tabletop interaction (Wilson, 2007). With a depth camera mounted above the tabletop, the depth image of the table surface could be monitored and reconstructed in real-time creating a physical interaction experience with a physics engine. An application of a car racing game was demonstrated where tangible objects could be used to cause physical interaction, such as collisions, with the virtual objects (See Figure 2.35).

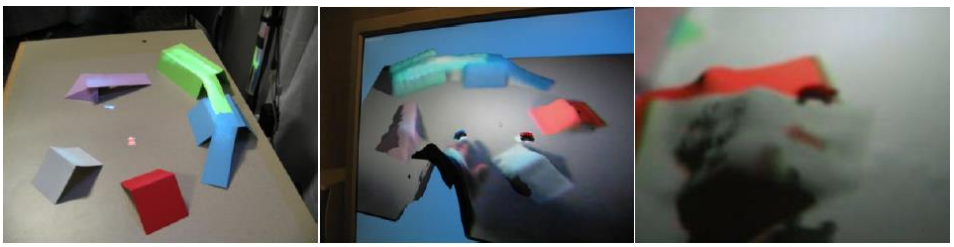


Figure 2.35: Projected race cars are racing around and onto real paper ramps (Wilson, 2007) (left), rendered view of the simulation (middle), car flies off the ramp (right)

Benko et al. presented MirageTable (Benko et al., 2012), a situated AR system that utilized a depth camera, a stereo projector, and a curved screen. MirageTable provided instant 3D capture of physical objects and the user, and rendered stereographic 3D content. The system also tracked the user's head and supported freehand interaction without external instrumentation. The authors presented applications in virtual 3D model creation, interactive gaming, and 3D teleconferencing (See Figure 2.36).

Figure 2.36: MirageTable offers remote collaboration in 3D (Benko et al., 2012), (left, middle left), digitizing any object on the surface (middle), freehand physics-enabled interaction (right)

2.4.1.3 Situated Room-scaled SAR

Recently, situated AR has been extended to operate beyond a tabletop to a wall and, eventually, the entire room. WorldKit (Xiao et al., 2013) offered a ubiquitous system that combined a depth sensor and a projector to turn any surface into a touch interface. They provided a framework for developers to easily develop interfaces anywhere in the room.

Jones et al. demonstrated IllumiRoom (Jones et al., 2013), a projection system that augmented the surrounding area of a television to enhance gaming experiences. Various methods of projected visualization were investigated, as well as effects such as changing the room's appearance, extending the field of view, inducing motion, and enabling new gaming experiences. The system

Figure 2.37: Illumiroom (Jones et al., 2013) enhances gaming experience by making use of space surrounding the TV.

could calibrate itself and a depth sensor provided scene spatial understanding, allowing projection distortion to be removed (See Figure 2.37).

RoomAlive (Jones et al., 2014) extended projection systems such as in Illumiroom to cover all sides of the room, enhancing immersion for the user. The system offered dynamic interactive projection mapping that adapted to any room. It supported multiple input actions including touch, shoot, stomp, dodge and steer. Each projector and depth camera pair was separately auto-calibrated and self-localized (See Figure 2.38).

Figure 2.38: RoomAlive (B. Jones et al., 2014) creates immersive experience by using multiple pair of projector-depth camera units for the entire room (left), shooting game (middle), dodging game (right)

2.4.2 Transparent Situated Display in Augmented Reality

HoloDesk (Hilliges et al., 2012) is a situated AR system that uses a half silvered mirror to create a see-through display, where the 3D graphics are spatially aligned with the real world. A depth camera was used to support direct tangible interaction between real objects and virtual contents. Natural hand grasping is also supported, using kinematic particles to represent the hand and contact

between the particles and the target object simulated force on the virtual objects (See Figure 2.39).

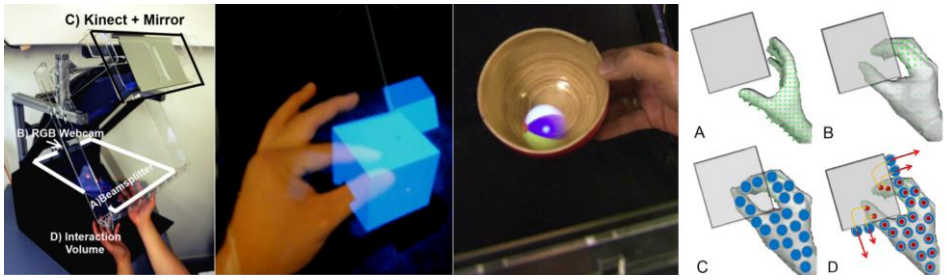


Figure 2.39: Setup of Holodesk (Hilliges, Kim, Izadi, Weiss, & Wilson, 2012) (left), hand grasping of a virtual cube (middle left), physics-based interaction with real object (middle right), hand is represented by kinematic

SpaceTop (J. Lee et al., 2013) integrated a transparent display and depth camera to extend the traditional desktop and create seamless 2D and 3D manipulation. It supports multiple modes of input including type, click, draw in 2D, and direct manipulation in the space above the keyboard using hand pinch detection (See Figure 2.40).

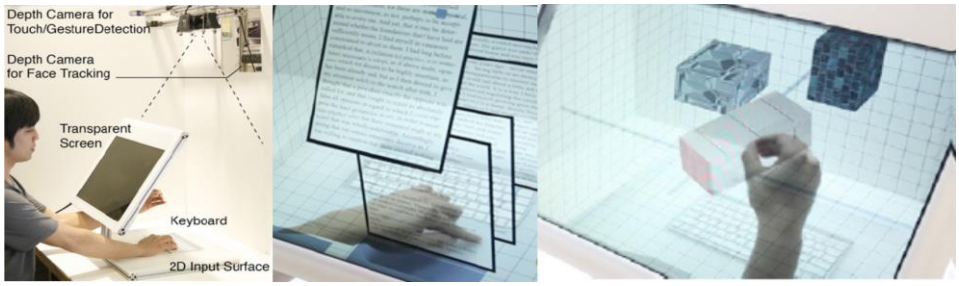


Figure 2.40: Setup of SpaceTop (J. Lee, Olwal, Ishii, & Boulanger, 2013) (left), 2D document browsing in SpaceTop (middle), 3D direct manipulation with pinch gesture (right)

2.4.3 Recent Trends in Depth Sensing for Handheld and Head Mounted Display

In this section, examples of recent advancement in depth sensing technology for handheld devices and head mounted display are presented in Section 2.3.2.1 and 2.3.2.2, respectively.

2.4.3.1 Depth Sensing in Handheld Devices

Google Project Tango (Project Tango) used customized hardware and software to track the motion of a handheld device in 3D to create a map of the environment (See Figure 2.41 (left)). The integrated sensors allow rapid 3D measurement of the surroundings, and the system can update its position and orientation in real-time. Another handheld device of note is Dell Venue 8 7000 Series (Dell Venue 8 7000 Series) as shown in Figure 2.41 (right), the first tablet with an integrated Intel RealSense depth camera. One sample application was in depth photography where a high-definition depthmap allows for measurement, refocusing, and selective filters.

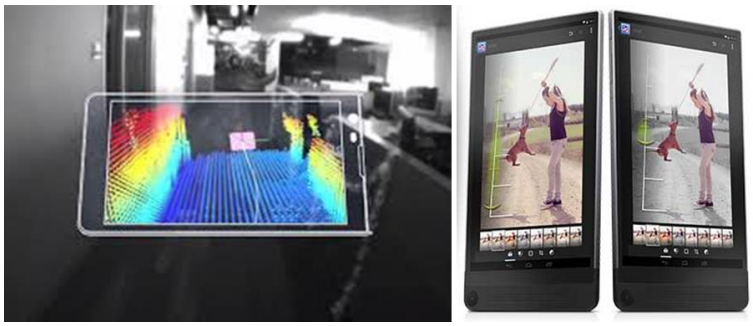


Figure 2.41: Project Tango shows scene tracking (Project Tango, 2015) (left), Dell Venue 8 7000 Series shows different filters applied to depth supported photograph (Dell, 2015) (right)

2.4.3.2 Depth Sensing in Head Mounted Display

Depth sensing is becoming a core component in commercial AR systems. Meta Glass (Meta) is an optical see-through HMD with an integrated depth sensor to support hand gesture input. With the current version of Meta Glass, the field of view (FOV) of the display and the gesture support are limited. The Leap Motion sensor can track hand poses and gestures with high fidelity, and Leap VR integrates this sensor with the Oculus DK2 (LeapVR, 2015), providing high DOF hand tracking in VR. The Oculus DK2 provides a wide FOV, however the stereo images displayed are based on IR illumination and can only be shown grayscale. The Nimble SDK is another hand tracking framework that uses a PMD depth camera to offer high DOF hand tracking. NimbleVR combines this SDK with the Oculus DK2 to provide a VR experience but not in AR (NimbleVR, 2015).

The Microsoft HoloLens device (Microsoft Hololens, 2015) is comprised of an optical see-through HMD with a built-in depth sensor for scene tracking, reconstruction, and hand gestures recognition, as shown in Figure 2.42d. Figure 2.43(left) shows an actor wearing HoloLens performing an "air tap" gesture for selecting a target from distance. Figure 2.43(right) shows a demonstrator holding a physical device that was used to activate a virtual shield.

Figure 2.42: (a) Meta Glass (Meta, 2015), (b) Leap VR (Leap Motion, 2015), (c) NimbleVR (Nimble VR, 2015), (d) Microsoft HoloLens (Microsoft HoloLens, 2015)

Figure 2.43: Microsoft HoloLens; air tap (left), Project X-Ray (right)

A concept video by Magic Leap (Leap, 2015), illustrates natural hand interactions with their AR interface. Figure 2.44(left) illustrates user's view through an optical see-through HMD where an actor reaches out to touch an icon on a floating menu interface. Figure 2.44(right) illustrates an actor reaching out and grab a 3D mail icon.

Figure 2.44: Magic Leap's concept video; touch (left), grab (right)

In addition to AR, there has been interest in natural interaction in the VR space as well. Recently, Oculus {Oculus, 2015} introduced Oculus Touch, a hand-held device that supports 6 *dof* tracking in space for a direct manipulation in VR. Figure 2.45(top) illustrates an actor wearing an Oculus HMD interacting in VR using a pair of Oculus Touch devices. Figure 2.45(bottom) shows Oculus Toybox, an immersive sandbox environment for collaboration in VR that supports physical interaction with virtual contents through Oculus Touch.

Based on these trends, the future of natural hand interaction for AR looks promising. Much of the development demonstrated by major developers has reaffirmed our confidence in natural hand interaction as a primary input for AR. We believe that this interaction technique will become commonplace as AR interfaces become pervasive.

Figure 2.45: Oculus Touch (Top) and Oculus Toybox (Bottom)

2.5 Shortcomings in Previous Research and Technical Challenges

In this section, we summarize the shortcomings of previous research and the technical challenges faced in Natural Interaction in AR. Following the description of each issue, we propose a methodology for how the issue could be resolved, and finally show how these relate to the subgoals of this thesis given in Section 1.1.

2.5.1 Need Insights into Characteristics of Affordance in an Environment Aware and Physics-enabled AR System

There has been little research into user's natural behavior when interacting with AR systems that are aware of physical changes to the environment and support physical interaction as if the virtual contents are real. One example of a system that functioned in this manner was by Wilson (Wilson, 2007) where he used a depth-sensing camera to reconstruct the surface of the interactive area of the table, however, 2D surface projection was used instead of 3D as in AR.

We can learn from users through observation during their interaction with these AR system. Findings from observation can help determine the characteristics of affordance and the user's expectation of such an AR interface, which in turn leads to insights into user's natural behavior and approaches that can be used to improve and enhance user experience through natural hand interaction. Since there is no available platform that supports these features, the resulting platform that we develop can be used by the other researchers to conduct studies for AR. This issue is addressed in Chapter 3: Natural Hand Interaction in Augmented Reality with Environment Awareness and Physics Simulation.

The corresponding subgoal to this issue is SG-2, *"Learn from users through observing their interaction with an AR system that offers environment awareness and physically-based interaction, then use this information to determine the characteristics of affordance of such an AR interface. The outcome of this goal is insights into user's natural behavior and approaches that can be taken to improve and enhance user experience through natural hand interaction."*

2.5.2 Limited Body of Knowledge on Natural Hand Interaction for AR

To date, there has been no in-depth research into user-centered design for natural hand interaction for AR, only research that explores multimodal gesture and speech interfaces where natural hand interaction is used as an auxiliary input to speech (Heidemann et al., 2004; Olwal et al., 2003). Other research provides simple gesture sets that are designed by the researchers for easy recognition as the technology permitted at the time of the research (Kolsch et al., 2006). There are also research papers that demonstrate physical interaction between natural hands and virtual objects through a simulated physical environment where the user can grasp onto the virtual object using simulated force and friction (Benko et al., 2012; Hilliges et al., 2012), which results in a trade-off between interactivity and precision of the direct manipulation through physical simulation.

A major shortcoming found in previous research is that only a small number of gestures are used and therefore only a limited number of tasks can be performed. Furthermore, those gestures were mainly designed for easy recognition and sometimes mapped arbitrarily to the given command with few agreements across interfaces. To solve this issue, a user-centered study is conducted to learn natural hand gestures that are preferable and easy to perform in AR. These findings can be used for future design guidelines. This study is presented in Chapter 4: User-defined Gestures for Augmented Reality.

The corresponding subgoal to this issue is SG-3, *"Learn hand gestures that are preferable and easy to perform in AR from users and create design guidelines from the findings. Successful completion of this goal will result in the first set of user-defined gestures for AR and the classification of those gestures into a gesture taxonomy for AR."*

2.5.3 Limited Technology for Natural Hand Tracking

Past gesture interfaces for AR could offer only a small number of hand gestures due to limited vision-based tracking technology when standard color cameras were used (Fernandes & Fernandez, 2009; Heidemann et al., 2004; Kaiser et al., 2003; Kolsch et al., 2006; Lee & Hollerer, 2007). The emergence of depth sensing technologies and better hand tracking algorithms has resulted in a much improved hand tracking software. High *dof* hand pose estimation is now achievable without the need for data gloves or external sensing. This allows for designing more complex natural hand interaction, however at the time of this research, there was no hand tracking software available for researchers. As a result, we developed a gesture interface that makes use of a depth sensor to perform hand pose estimation and tracking in real-time. We share our findings from the development process in Chapter 5: Lessons Learnt from the Development of Gesture Interface for Augmented Reality.

The corresponding subgoal to this issue is SG-4, *"Develop a gesture interface that utilizes depth sensing technology for hand tracking and recognition. The outcome of this goal is an interface which supports novel natural hand interaction techniques."*

2.5.4 Lack of Platform to Support Natural Interaction for AR

At the time of this research, there was no software framework that supported natural interaction integrating high degree-of-freedom natural hand interaction, and gesture and speech interaction as the primary inputs. Moreover, there was no suitable hardware display that could provide an immersive AR experience through a wide field-of-view screen.

As a result we developed a software framework and a custom hardware solution to address these issues. Our solution is covered in Chapter 6: Multimodal Augmented Reality Framework and Interaction Techniques.

The corresponding subgoal to this issue is SG-5, *"Develop an AR framework that supports natural interaction as the primary inputs, in this case, a direct natural hand interaction and an indirect multimodal gesture and speech interaction. The success of this goal is based on a working and demonstrable system implemented using this framework."*

2.5.5 Need for Formal Evaluation to Compare Direct Natural Hand Interaction and Indirect Multimodal Gesture-speech Interaction

Past research has demonstrated both cases of gesture-only and gesture-speech interface for AR. Studies were conducted to compare the usability between gesture-only and multimodal interface (M. Lee et al., 2013), however it is still unclear what are the strengths and weaknesses of each techniques and when the system should offer one interaction techniques over another. It is crucial to know the differences and benefits of these interaction techniques.

In this research we develop and compare two natural interaction techniques: a novel direct natural hand interaction technique and an indirect multimodal gesture and speech interaction technique. Both techniques are designed based on experimental findings of our guessability study and past guidelines. We validate these techniques in terms of performance, usability, and user preference. We present findings that show that the directness of interaction influences user preference in choosing different interaction techniques for different tasks. This study is covered in Chapter 7: Comparison Study between Direct and Indirect Natural Interaction in Augmented Reality.

The corresponding subgoal to this issue is SG-6, *"Evaluate and compare two natural interaction techniques, a novel direct natural hand interaction technique and an indirect multimodal gesture and speech interaction technique. The success of this goal is based on whether there are differences between the two interaction techniques in term of performance, usability, task load, and user preference. Moreover, the strengths and weaknesses of each technique should be identified."*

2.6 Conclusion

In this chapter, we reviewed related works that are relevant to our research. We gave an overview of AR in term of definition, software architecture, applications, evaluation methods, and AR for collaboration. We explained user interfaces in regard to direct manipulation and intelligent agent. Intelligent user interface was covered and the two types of an adaptive direct manipulation interface, which are adaptive user interface and multimodal user interface, were presented. An in-depth review into interaction techniques in AR prior to an introduction of consumer depth sensor had been presented. This included tangible AR, hand gesture, haptic, speech, and the combinations of these inputs. We then gave a concise explanation and example research for environment awareness and physically-based interaction in AR.

Next, we reviewed recent research that utilized depth sensing for interaction and scene understanding. Consumer depth sensors and natural interaction platforms were shown. The usage of depth sensing to solve hand pose estimation and recognition problem was discussed. Research into natural hand interaction and gestures performed on and off the interactive surface area was considered. AR research that combined depth sensing and different display technologies,

including spatial AR using projectors and transparent situated displays, and state of the art depth sensing in handheld and HMD configurations, were presented.

Finally, we summarized the shortcomings and technical limitations in previous research that we set out to address. It can be seen that much effort was put into enabling the AR systems, while there is still limited understanding of user experience and preference interacting in AR. This is the motivation of this research that is to better understand these aspects of interaction in AR particularly using natural hands.

Part II
Exploring and Understanding Natural Hand Interaction in Augmented Reality

Chapter 3
Natural Hand Interaction in Augmented Reality with Environment Awareness and Physics Simulation

In augmented reality (AR), one approach commonly used to support three-dimensional (3D) manipulation of virtual objects is the integration of a Tangible AR interface (Billinghurst et al., 2008), Tangible AR interfaces exploit a user's natural experience interacting with physical objects in the real world to make interacting with virtual objects more intuitive and natural. A downfall of this technique is that users often expect all actions that they would normally apply to physical objects to be supported by the system, which is often not possible (Hornecker & Dünser, 2009).

One example of this is shown with interaction based using a paddle which consists of a fiducial marker with a handle. The paddle can be used for positioning virtual objects in AR (Irawati et al., 2006), as shown in Figure 3.1. By touching the virtual object with the paddle, the virtual object is selected and positioned onto the paddle. By tilting the paddle, the object is placed back into

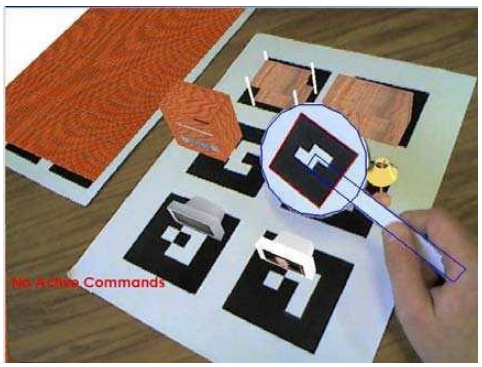


Figure 3.1: The act of tilting the paddle, place the virtual object in the scene but the experience is not seamless when the object does not react to friction and gravity when the paddle is tilted (Irawati et al., 2006)

the scene. Although this tilting gesture is intuitive, the virtual object does not behave as one would normally expect a physical object to do so, in this case, to slide off due to gravity when the paddle is tilted. Instead, the object is just placed onto the desk below the paddle, lessening the realism of the interaction.

To improve an AR experience, virtual content should behave realistically in the physical environment it is placed in. Computer vision based fiducial marker and natural feature registration algorithms are able to calculate the pose of a given target, but have no awareness about the environment the target exists in. This lack of awareness can cause the virtual content to float above real objects or appear inside them, or occlude objects that they should appear behind, breaking the illusion that the virtual content exists in the real world.

Environment awareness is a term that describes the awareness of physical changes in the scene in real-time permitting interaction between physical and virtual objects through physics simulation, known as physically-based interaction. As people commonly physically interact in the real world using their hands and this technique allows direct natural hand interaction with virtual content, it empowers users to seamlessly interact in both real and virtual worlds bridging the gap between the two.

In this chapter, an exploration into natural hand interaction in an environmentally aware physics-driven simulation is presented. Two AR systems utilizing depth sensors were developed, with the aim of offering a natural and intuitive way to interact. These systems permit us to create systems that demonstrate the advantages of having physically-based interaction and environment awareness. The depth sensor is used to examine the 3D interactive space, and by knowing the transformation between the depth camera and the AR viewing camera, virtual content can be realistically composited in the environment. Users can interact with the virtual content in a natural and intuitive way using their hands and other physical objects.

The first system was designed to support face-to-face collaboration in a tabletop setting, such as in an office, and is presented in Section 3.1. Section 3.2 presents the second system that was designed for a mobile tablet setting, and permits ubiquitous interaction with the surrounding environment, for example living room space in a gaming scenario. Section 3.3 presents observations and feedback from public demonstrations and summarizes the characteristics of affordance of the AR systems presented. Section 3.4 summarizes lesson learned from the two AR systems. The chapter is concluded in Section 3.5.

3.1 AR System for Face-to-face Collaboration on a Tabletop

3.1.1 System Overview

An overview of the system is presented in Section 3.1.1.1, followed by the implementation in Section 3.1.1.2 where the key components are covered including marker tracking, depth acquisition, image processing, physics simulation, communication and rendering. Finally, the natural hand interaction is explained in Section 3.1.1.3.

3.1.1.1 Setup and Architecture

3.1.1.1.1 System Setup

To create an interaction volume, a Microsoft Kinect is positioned above the desired interaction space facing downwards, as shown in Figure 3.2. A printed reference image marker is placed in the interaction space to calculate the

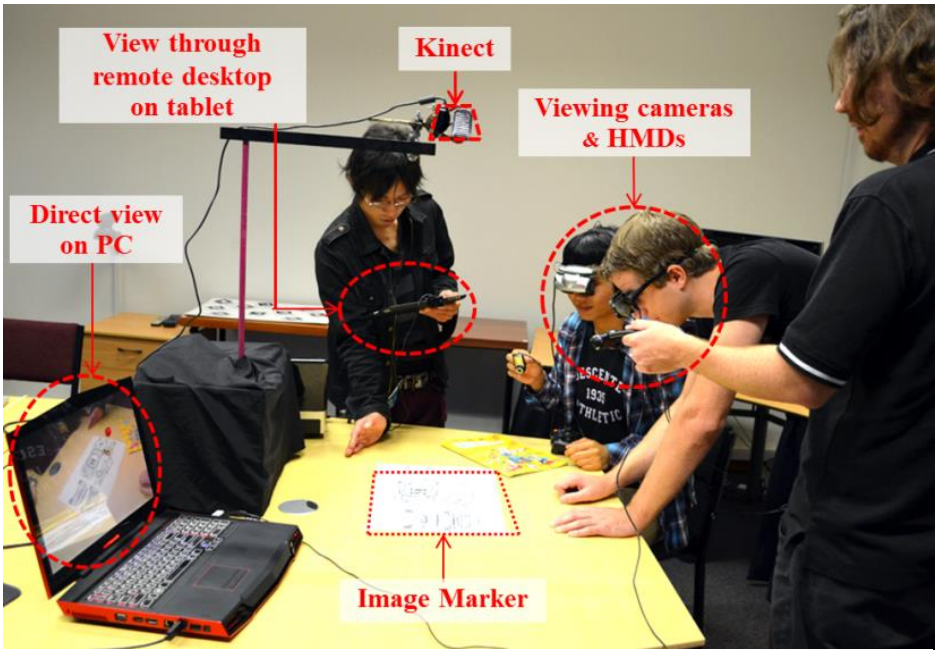


Figure 3.2: System setup for face-to-face collaboration on a tabletop

transform between the Kinect coordinate system and the coordinate system used by the AR viewing cameras. Users can use several types of displays connected to the client PC for viewing the AR content, such as a handheld or head-mounted displays, or a fixed monitor.

3.1.1.1.2 ARCHITECTURE

The system uses a client-server model to offload rendering from the server. The server process is comprised of five stages, while the client has three stages as shown in Figure 3.3.

3.1.1.2 Implementation

The components of the system are described in greater detail in the following sections, and the data flow process within the system is illustrated in Figure 3.4.

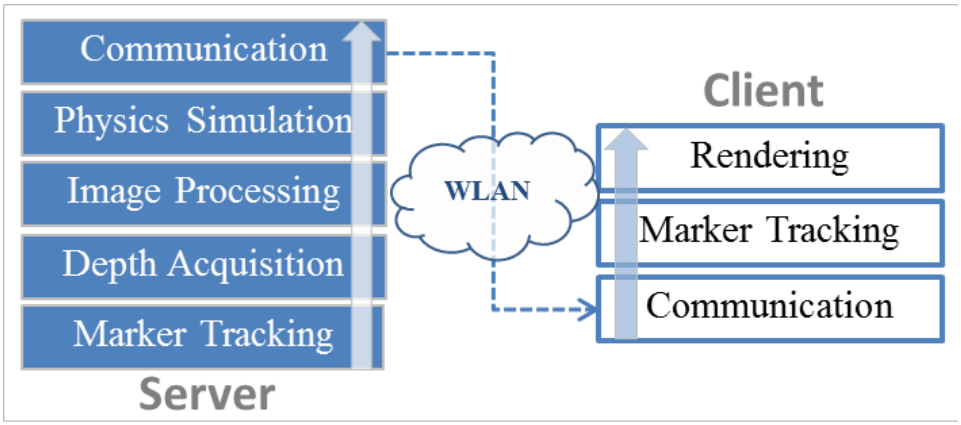


Figure 3.3: Processes overview of collaborative AR system

3.1.1.2.1 MARKER TRACKING AND DEPTH ACQUISITION

The OPIRA natural feature tracking library (Clark et al., 2008) is used for natural feature based registration due to its robustness to perspective distortion and other image transformations. Due to their fast computation time, SURF (Bay et al.,2006) is used as the feature descriptor. The OpenNI library (OpenNI, 2015) is used to interface with the Kinect. Color and depth images are accessed through the API, and the two images are automatically transformed and rectified.

During the initialization phase, the color and depth images from the Kinect are aligned using OpenNI (see Figure 3.4). The transformation from the image marker to the Kinect is calculated using natural feature based registration from OPIRA. The homography of the marker from the Kinect's viewpoint, H_{Kinect}, is calculated and the four corners of the marker are projected into the 2D Kinect's color and depth images. The 3D position for each corner is calculated using the OpenNI function, *ConvertProjectiveToRealWorld( )*, which converts the (u_0, v_0) 2D point in pixel units in the input image into a 3D point (x_0, y_0, z_0), in the real world coordinate system.

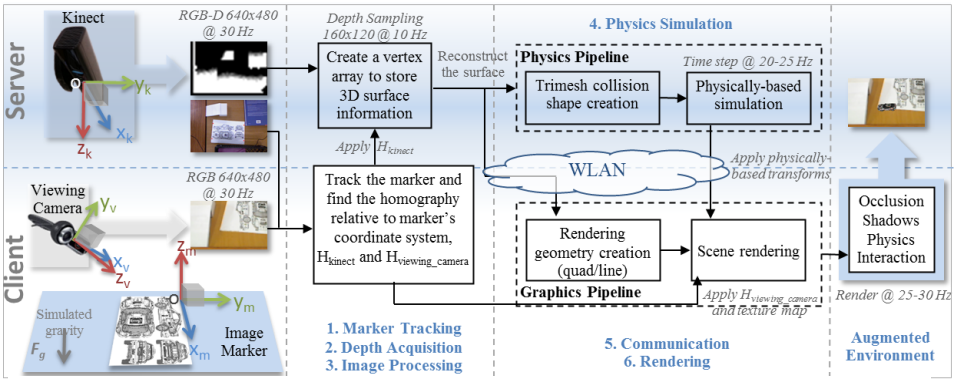


Figure 3.4: Overall illustration of our AR system's architecture showing coordinate systems and information processing in each stage of the process in both server and client

The input image from the viewing camera is processed to find the marker homography from the viewing camera, $H_{viewing\_camera}$, and the size of the marker is calculated in millimeters using the Euclidian distance between corners. The AR coordinate system is established with the origin at the top left corner of the marker, and this coordinate system is applied to the camera in the OpenSceneGraph scene. Finally, the transformation between the corner positions in the Kinect coordinate system and the corner positions in the AR coordinate system is calculated and stored.

With the transformation between the Kinect and the AR coordinate systems known, 3D data from the Kinect can be transformed into the AR coordinate system (Figure 3.5a). Assuming the plane that the marker lays on is the ground; objects can be segmented by using a simple threshold of the distance of each pixel from the ground plane. Figure 3.5b shows the point cloud data captured by the Kinect projected into the AR coordinate system as a mesh.

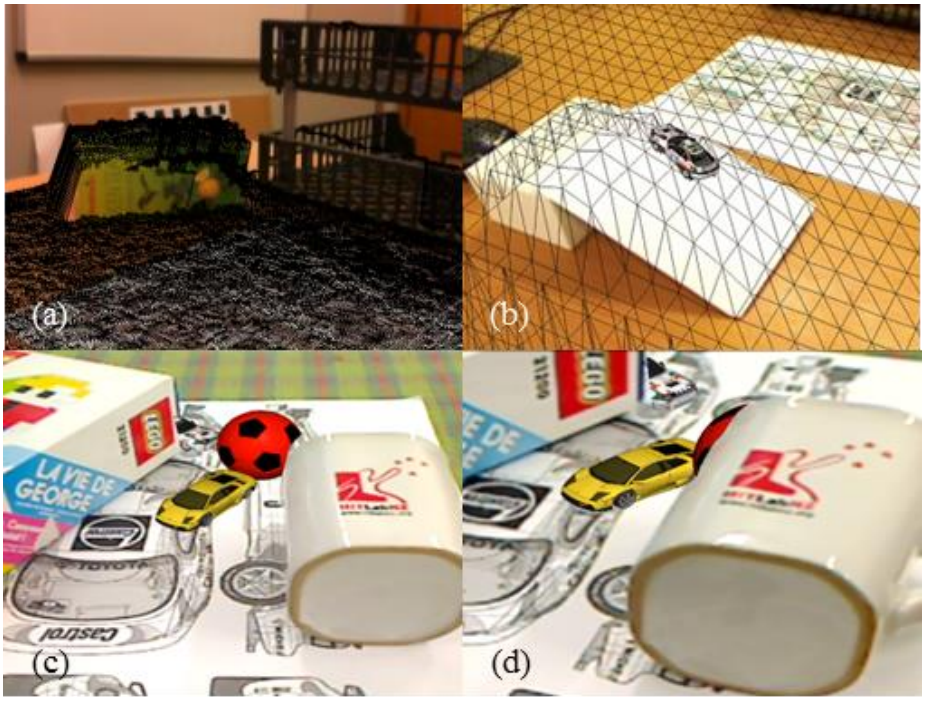


Figure 3.5: (a) Point cloud (b) Wireframe (c) and (d) Occlusion from different viewpoints

3.1.1.2.2 IMAGE PROCESSING

The depth image obtained by the Kinect is prone to missing values due to shadowing of the infrared data. To resolve this, missing values are identified and an Inpainting algorithm is used to estimate their values. The OpenCV function, cvInpaint(_), is used with Navier-Stokes Inpainting method for this purpose.

The depth image is resized from 640x480 resolution to 160x120 using the nearest neighbor interpolation through OpenCV's function, cvResize( ). The coordinate system of the Kinect, (x_k, y_k, z_k), is aligned to the image-based coordinate system, (x_m, y_m, z_m), such that the upper left corner of the marker represents the origin in both the real and the virtual world. The depth information from the 160x120 depth image is stored in a vertex array at

millimeter scale. This vertex array represents the spatial information of the surface of the interaction space above the tabletop. The origin depth is set to zero, so any points above registration marker will have positive depth values and those that are lower will be negative.

3.1.1.2.3 PHYSICS SIMULATION

The data in the vertex array is used to reconstruct a physical proxy of the table surface. The proxy is created as a triangle mesh (trimesh) with the Bullet Physics Engine's function btBvhTriangleMeshShape( ). The trimesh represents the terrain on the tabletop and is used by the physics engine for collision detection. Optionally, the trimesh can be rendered using Delaunay triangulation in the AR view to show the physical representation of the world, as shown in Figure 3.5b.

The system assumes that the registration marker lies flat on the tabletop and the simulated force of gravity is directed down perpendicular to the marker. To allow for additional configurations, the Kinect's built-in accelerometer could be used to determine the Kinect's orientation in the real world, and this could be used to align the direction of the physical simulation's gravity to the direction of the real world's gravity.

The trimesh in the physics simulation is updated at 10 Hz, which allows the user to interact with the virtual content with realistic physics. For example, the user's hand or a book can be used to pick up or push virtual objects around. However, this interaction is limited, especially for more complex shaped objects, due to the single point of view of the Kinect. More realistic interaction would require multiple views or tracking of 3D objects to determine the orientation and complete shape of the object.

3.1.1.2.4 COMMUNICATION

The VRPN library is used to create network connections between the server and clients. The communications are unidirectional, transmitting data from server to client. There are two VRPN classes used, tracker and image server. The tracker is used to transmit the pose of objects and the image server is used to transmit the 3D surface of the ground plane.

For each type of virtual object in the simulation, a corresponding class based on the *vrpn_Tracker* class was implemented. A VRPN tracker object can have multiple sensors, with each sensor having attributes such as position and orientation. An example would be a tracker object representing a car might have a chassis and four wheels, with the pose of each component represented by a sensor. The surface information is encoded into a 2D array of floating point number and stored in a *vrpn_Imager_Server* object.

Each update, the communication transmitter queries the physics world for the current pose of each object as well as the surface information and encodes them into the packet to send over the network. Once the client receiver receives the packet, the transformations are applied to each object and the ground mesh is updated.

3.1.1.2.5 RENDERING

OpenSceneGraph, a 3D graphics API that utilizes a tree structure called a Scene Graph to organize the scene, is used for rendering. The dynamic transformation of the physical proxy in the simulated world can be easily applied to the scene graph for visual feedback. The input video image is rendered first, with all the virtual objects rendered on top. At the top level of the scene graph, the AR viewing transformation is applied for all virtual objects so they appear anchored in the real world. The terrain data is rendered as an array of quads, with an alpha

value of zero. This allows realistic occlusion effects of the terrain and virtual objects, while not affecting the users' view of the real environment, as shown in Figure 3.5c and 4.6d.

One limitation of OpenSceneGraph is that, planes with an alpha value of zero cannot have shadows rendered on them, so a custom fragment shader was written which allows shadows from the virtual objects to be cast on the invisible terrain map. The shadow casting can be seen in Figure 3.5c and 3.5d. The shadows add an additional level of realism, as well as important depth cues which allow users to intuitively understand the distance between virtual objects and the ground plane.

3.1.2 Natural Hand Interaction for Collaborative Platform

To allow interaction with the virtual content natural hand interaction was chosen as the primary user interface. People are used to interacting with real objects using their hands in many tabletop based tasks, and activities such as collaborative design and gaming can directly benefit from the ability to use natural hand input.

To simplify the design of the physically-based interaction using natural hands for this system, we made the assumption that the user's hands will not completely overlap each other, which would block the Kinect's view of a hand interacting with a virtual object. We believe this assumption is fair considering typical user behavior and use of space around a tabletop setting during collaboration (Scott & Carpendale, 2010).

Three methods were investigated for physically-based natural hand interaction, a trimesh-based approach, direct sphere particle-based substitution and variational optical flow. All three methods share the same initial three steps, with additional forth steps for direct sphere substitution and variational optical flow. The following sections discuss these processes.

3.1.2.1 Hands Segmentation

Step 1: *Threshold the color image at the marker depth.* We assume that the marker lies on the table surface and all hand interactions are performed on and above the table surface. This helps remove skin-color pixels in the background that do not belong to a hand. Once the marker's depth is obtained from the Kinect, a threshold value of 5 mm above the marker's depth is applied to the depth image and a binary mask is created. The mask is applied to the color image and the result is shown in Figure 3.6b.

Step 2: *Find skin color pixels.* Skin and non-skin color probability tables created using Gaussian Mixture Model for RGB color space provided by (M.

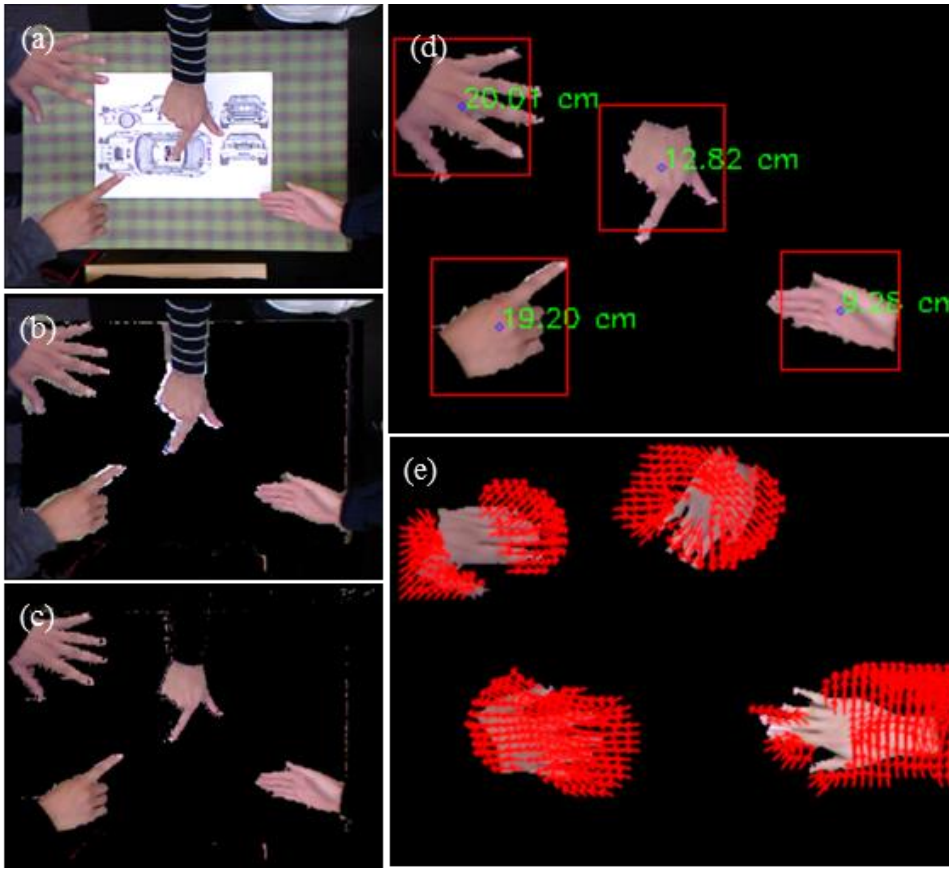


Figure 3.6: (a) Input image (b) Depth threshold (c) Skin color segmentation (d) Hand contour and consistent bounding square (e) Variational optical flow

Kolsch & Turk, 2004), are used. The skin color pixels can be determined by the ratio, r_p, between skin probability, P_{skin}, and non-skin probability, $P_{non\text{-}skin}$, compared to a threshold value, ϕ. Experimentally, we found a ϕ value of 0.2 worked well. A binary mask is created as shown in Equations 3.1 and applied to the color image as shown in Figure 3.6c.

$$r_p = \frac{P_{skin}}{P_{non-skin}} \quad and \quad \text{Pixel}_{\text{Mask}} \begin{cases} 1 & r_p \geq \phi \\ 0 & r_p < \phi \end{cases} \tag{3.1}$$

Step 3: *Apply a connected component filter and find hand contours.* After Step 1 and 2, we assume that every remaining contour, *c*, can be considered a hand contour, C_{hand}, if its perimeter, ρ_c, is larger than the sum of the width, I_w, and height, I_h, of the image over the specified scale, P, as defined in Equation 3.2. The perimeter scale used is $P = 4$. Further discussion of connected components can be found in (Bradski & Kaehler, 2008).

$$\forall c \in C_{hand} : \rho_c \geq \frac{\sum I_w + I_h}{P} \tag{3.2}$$

3.1.2.2 Mesh-based Representation

With the segmented hand image, a new trimesh can be created and overlaid on top of the ground mesh (Figure 3.7a). The area of the ground mesh that is occluded by the hand is not updated. By representing hands with a second mesh that updates more frequently than the ground mesh, the accuracy of the physics simulation is increased, the stability of the ground mesh is improved and the possibility of virtual objects falling through the ground geometry is reduced.

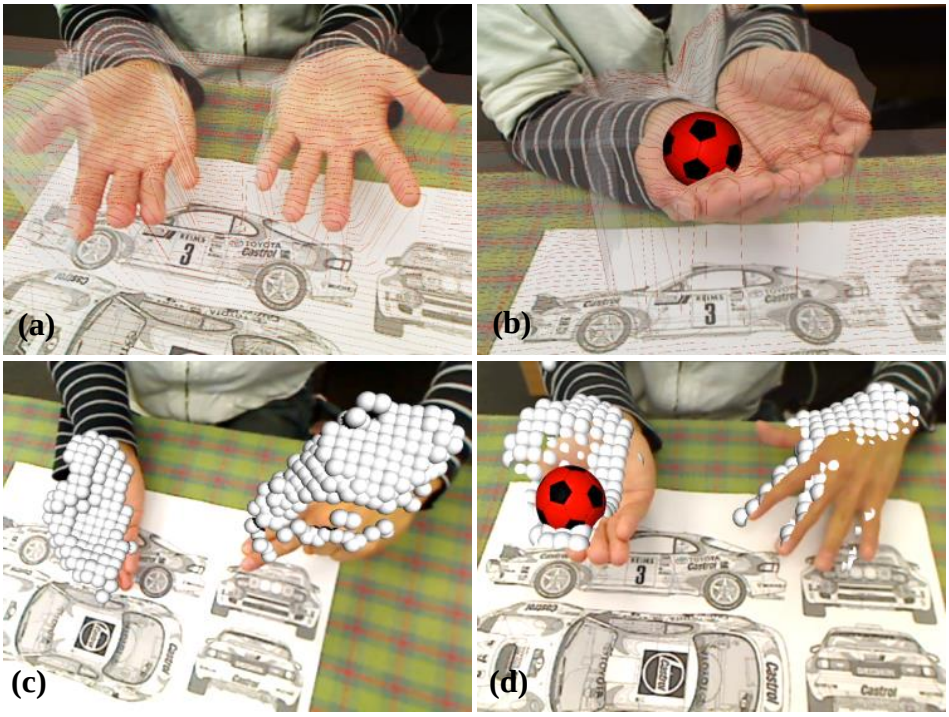


Figure 3.7: (a) and (b) Mesh-based representation (c) and (d) Particle-based representation with direct sphere substitution

The advantage of a mesh-based representation is that number of hands present in the scene does not affect the system performance. The main disadvantage of this method is that the single viewpoint of the Kinect means that the reconstructed mesh occupies the volume under the hand as well.

3.1.2.3 Direct Sphere Substitution

Step 4 (a): *Find bounding squares for hand contours*. The default bounding box of each contour can be obtained directly in OpenCV, however, a consistent square that changes relative to the depth of each hand is required. To achieve this, we calculate the center of hand contour using spatial moments and create a square for each center, as shown in Equation 3.3. The square's width is fixed using an average hand size, $w_{cm} = 25\ cm$, and the corresponding square width in pixels, w_{pixel}, is calculated as shown in Equation 3.4, where h is the height of the

hand above the ground plane, m_{pixel} is the marker's width in pixels, m_{cm} is the marker's width in centimeters, and $\tau = 0.9$ *is a* constant for scaling the height. The values in the bounding square in Figure 3.6d show the estimated height of the hand above the marker.

$$x_{center} = \frac{m_{1,0}}{m_{0,0}},\ y_{center} = \frac{m_{0,1}}{m_{0,0}} \quad and \quad m_{p,q} = \sum_{i=1}^{n} I(x,y)x^p y^q \qquad (3.3)$$

$$w = floor\left(w_{cm} \times \frac{m_{pixel}}{m_{cm}} + h \times \tau \right) and\ w_{pixel} \begin{cases} w & w_{odd} \\ w+1 & w_{even} \end{cases} \qquad (3.4)$$

The hand is represented in the physics simulation using an *N* by *N* grid of spheres where *N* is the grid resolution and must be an odd number (Figure 3.7c). This is achieved by dividing the bounding square to an *N* by *N* grid, and interpolating heights fora sphere at each grid point. A median filter is applied to remove outlying height values. Each sphere has a distance z, perpendicular to the ground plane, and the whole grid is positioned along the x-y plane according to the center of the bounding square. A sphere is excluded from collision detection if its value is outside the range of $z_{central} \pm Z$, where $z_{central}$ is the height at the center of the bounding square and *Z* is a cutoff value.

Computation time depends on the sphere grid resolution and the number of hands present in the scene. The resolution of the grid can beset to a lower value to improve performance or a higher value to improve accuracy. Experimentally we determined the optimal range of $N = 15$ to 29.

3.1.2.4 Variational Optical flow

Step 4 (b): *Estimate the optical flow.* Optical flow gives us the motion of image pixels on the x-y plane, and the corresponding change in depth value, δz, can be found from the (x, y) pixel in the depth image. For variational optical flow, a Combined Local-Global (CLG) optical flow method using a bidirectional multi-

grid solver (Bruhn, Weickert, Feddern, Kohlberger, & Schnorr, 2005) was utilized. CLG takes advantage of the global Horn-Schunck and local Lucas-Kanade optical flow method (Figure 3.6e). CLG optical flow requires three parameters: α, the regularization parameter that is a positive number which steers the smoothness of the flow field; ρ, the standard deviation of the convolution of motion tensor with Gaussian kernel which we set to 2.8;, and σ, the standard deviation of the convolution of preprocessed image sequence with Gaussian kernel, which we set to 1.5. These values were obtained experimentally, and found to give the best result with regards to image resolution of the hands and system performance.

For our system, spherical proxies were created for each skin pixel in a 160 by 120 image. Unfortunately, we found this method to be extremely computationally demanding, which lowered performance and made interaction difficult. While Variational optical flow is a promising method for particle-based representation, further optimization by down sampling the skin pixels or performing parallel computing on a GPU is required for real-time performance.

3.1.3 Performance

The goal of this research is to create a more realistic and engaging AR experience. To achieve this, the system must be capable of running in real time while ensuring the virtual content behaves realistically in the environment which it is in, and that the user can interact with the system in a natural and intuitive manner.

On an Intel 2.4Ghz quad core desktop computer, the applications are capable of running at over 25 frames per second (FPS) on the client. The server updates the interaction volume at around 11 FPS in the mesh-based and direct sphere substitution simulations and at 5 FPS in the variational optical flow simulation.

The Kinect error is less than half a centimeter at the ground plane when placed between 70 - 120 centimeters from the ground plane.

3.1.4 Applications

Two novel applications were developed to illustrate the potential of this system and for demonstrating to the public to obtain user feedback. The first application, ARMicroMachines, is an AR car racing game that allows user to use a Microsoft Xbox 360 game controller to drive virtual cars on the tabletop which interact with real world objects in real-time. By leveraging the information provided by the depth sensor, users can create an obstacle course using real objects. The second application, ARSandbox was developed to showcase natural hand interaction where users can interact with virtual objects that come in different shapes and sizes such as balls, boxes, pyramids etc. Further details of the feedback and suggestions for improvement are discussed in Section 3.3.

3.1.4.1 ARMicroMachines and Findings from Demonstrations

In ARMicromachines, the cars are represented in the physics world using the Bullet Physics Engine's raycast vehicle object, btRaycastVehicle, which provides physical attributes such as tire friction, engine and breaking forces, and suspension characteristics for each car to increase realism (see Figure 3.8). A single RGB camera is used as a viewing camera, and a projector provides a shared large screen display, as shown in Figure 3.9.

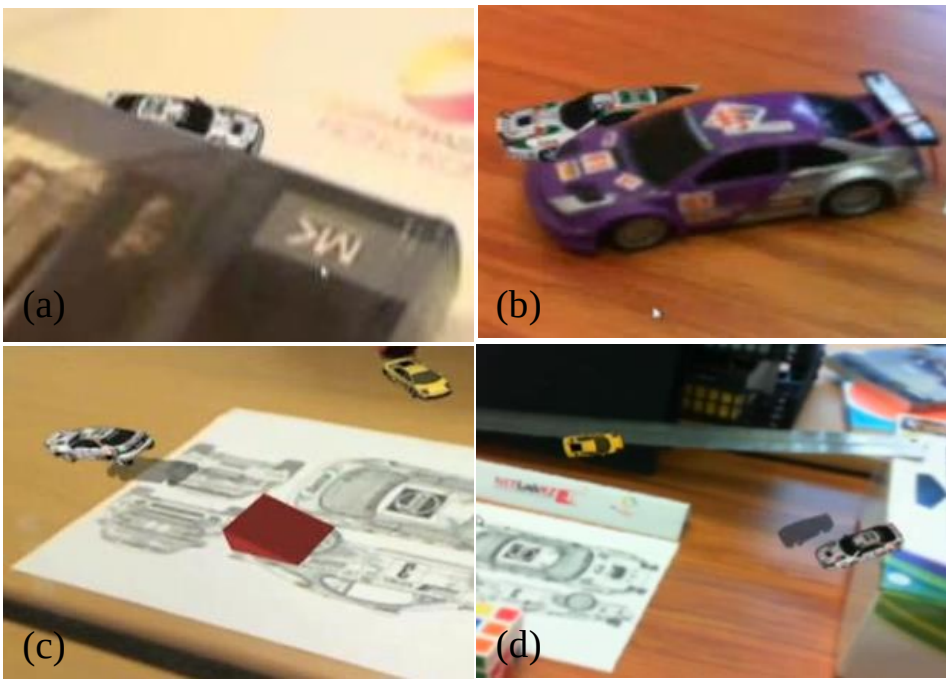


Figure 3.8: (a) The virtual car is occluded by a real book and (b) occluded by a real toy car (c) The car is flying off a red virtual ramp and cast a shadow on the ground and (d) falling off a real box and cast a shadow on the floor

ARMicroMachines was demonstrated to the public on several occasions including SIGGRAPH Asia'11 conference in Hong Kong, ISMAR'11 conference in Switzerland, and the CHINZ'12 conference and HITLab NZ open day in New Zealand. Hundreds of people tried the application and gave useful feedback providing the insight into how the application and the system could be improved. The age range of the users was from elementary school students to adults in their 70s.

During the demonstrations, users were given a general description of the system and were shown how to use the Xbox 360 game controller to control the car. Of these users, their experience with AR ranged from minimal to expert. While a formal evaluation was not conducted, a number of interesting qualitative observations were made during the demonstrations.

Figure 3.9: Public demonstrations of ARMicroMachines (Top) At HITLab NZ Open House 2011 (Bottom) At SIGGRAPH Asia 2011 Emerging Technology

With only minimal explanation, users were able to engage in the game quickly. Some groups designed their own tracks and challenged their friends to races. Others examined the interaction and tested the limitations of the system by directly manipulating the car with objects and their hands.

Users found the game play unique and fun. They were impressed with the realistic physics, which was possible with the environment awareness afforded by depth information. They expressed that they really liked the idea of the application, and could immediately see the value as an entertainment application. They liked being able to interact directly with the virtual cars, and could easily modify the environment that the cars were in by using real objects. Even novice AR users were able to intuitively understand how the interaction area could be changed by placing real objects within the view of the depth camera.

While users who were familiar with console based racing games were able to immediately understand the controls, some users who were not familiar had some initial difficulties controlling the cars. While this could be remedied by changing the control system or introducing new devices with different control metaphors (e.g. accelerometer or touch screen based interaction on smart phones), this may just be a required learning process for users.

The most common difficulty users had was controlling the vehicles due to the third person view of the cars affecting their spatial co-ordination, and many users expected the controls to map to the directions from their view point. For example, when the car was facing the user, pressing left on the control stick would cause the car to turn towards the user's right (the car's left), rather than to turn towards the user's left. To resolve this, a separate control scheme could be developed which drives the car with respect to the perspective of the user.

3.1.4.2 ARSandbox

AR Sandbox was developed to illustrate natural hand interaction in an unconstrained environment, as shown in Figure 3.10. Users are free to interact with the virtual objects provided including balls, boxes etc. By permitting customization of the physical properties of the simulated world such as gravity and friction, users can create their own games. For example, the tabletop could be turned into a soccer field using real objects to create the field perimeter and goals on each side. Players can use their hand to gently dribble the ball and shoot at the opposition goal. By lowering the friction of the ground, user can play air hockey with virtual objects, using a hand or other tangible object as a mallet, or simulate an environment in space or on the moon.

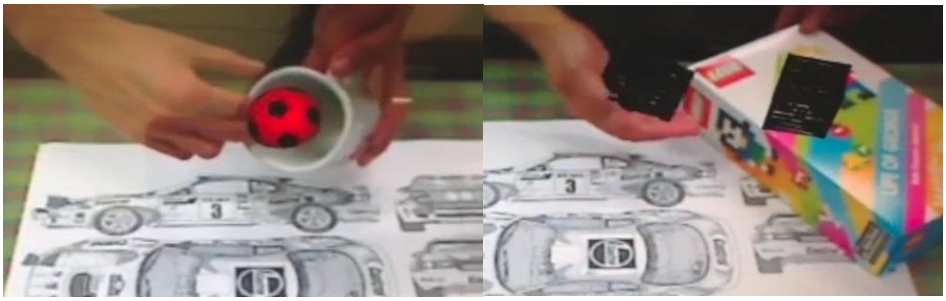


Figure 3.10: Virtual objects interact with natural hand and physical object in ARSandbox

Users reported that they found ARSandbox natural, intuitive, and fun. However, as the interaction was limited to only simple physical actions and the precision was limited users expressed the desire to see more ways to interaction with the virtual content with higher precision such as being to grasp a box and precisely stack it on top of another box.

3.2 AR System for Mobile Platform using Magnetic Tracking and Depth Sensing

Due to the high computational requirement and the use case scenarios of AR, past research has focused on face-to-face collaboration in AR in a tabletop setting, as was explored in Section 3.1. However, with rapidly improving processing power and increasing ubiquity, AR is also a promising technology to provide new experiences in the area of gaming, especially for console and mobile devices.

Recent developments in both PC and console gaming platforms have seen a common movement into using a tablet form factor with touch sensing capability to deliver an experience that offers a balance between screen size and mobility. For example, Sony has released the PlayStation Vita, which connects to the Playstation 3, the latest Nintendo Wii U system has the Wii U GamePad controller, and Razer has released the Razer Edge for PC.

With this new platform comes the opportunity to explore new AR interface and interaction techniques. To enable this, it is necessary to provide a combination of hardware and software that game and interaction designers can use to explore these new possibilities. Although advanced algorithms that unify tracking and reconstruction and offer unique interaction already exist, for example KinectFusion (Newcombe et al., 2011), they are often too computationally intensive for the current generation of mobile devices.

The focus and primary contribution of this section is exploring a new AR interface and utilizing the techniques of physically-based interaction, environment awareness, and natural hand interaction in this new setting of ubiquitous computing. We present a new platform named KITE which offers a fusion between software and hardware implementation, is simple to assemble, efficient to execute, and offers multiple interaction methods. In the following sections, the KITE platform is presented, beginning with hardware assembly and software architecture in Section 3.2.1. In Section 3.2.2, the four modalities of hand interaction are presented. The performance is discussed in Section 3.2.3 and the demonstrations that have been developed on this platform are described in Section 3.2.4.

3.2.1 System Overview

In this section, the hardware assembly of KITE is described, an overview of the software architecture is given and finally possible interactions are illustrated.

3.2.1.1 Hardware Assembly

The KITE platform is a mobile solution that can be quickly assembled from existing hardware. This is achieved by using a combination of off-the-shelf products (See Figure 3.11 for the listed equipment):

- Razer Hydra: A magnetic tracker for PC gaming.
- Primesense Carmine 1.09: A short range depth camera.
- Microsoft Surface Pro: An Intel powered Windows 8 tablet.
- Plastic Casing: A hand-made case from Polymorph plastic with a metal wire skeleton.
- USB Hub: To connect both tracker and camera to the tablet

Figure 3.11: KITE's components

Two Hydra controllers are attached to the tablet using the plastic casing, at a distance of 10 cm to minimize electro-magnetic interference with the tablet. The depth camera is attached to the top of the case and aligns with the controller's axis.

3.2.1.2 Software Architecture

The system architecture, shown in Figure 3.12, is comprised of four stages; point cloud acquisition, transformation, reconstruction and simulation. A point cloud

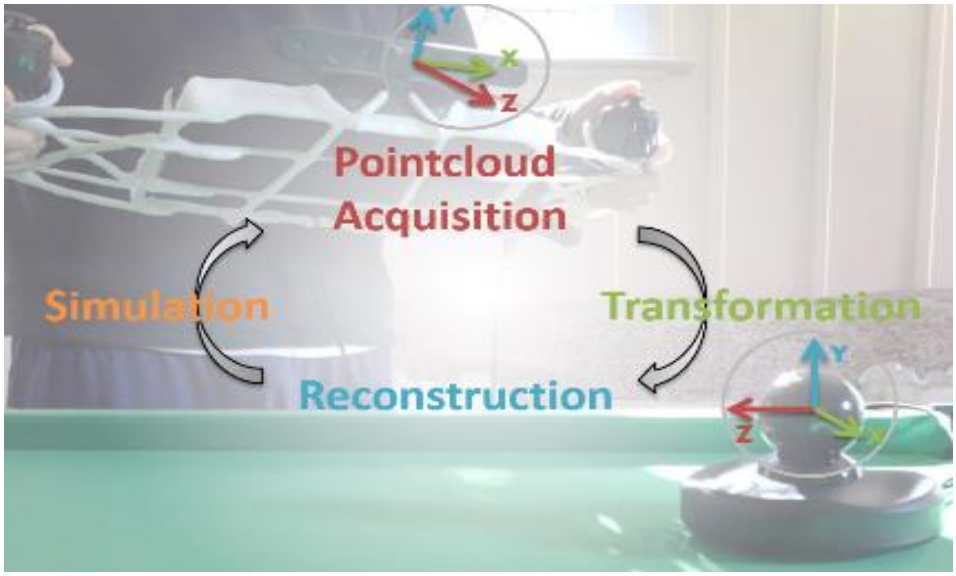


Figure 3.12: KITE software processes

is acquired from the depth camera using the OpenNI library. All points are transformed from the depth camera's co-ordinate system to the magnetic tracker's co-ordinate system. The transformed points are triangulated to create a vertex array for use in the physical simulation, which is handled by the Bullet physics engine.

To reduce the computational load on the device, three rules were established to determine when to update the environment model which is stored internally as a mesh. First, the frame-to-frame change in controller or transmitter position and orientation must not exceed a threshold, limiting updates to only occur when the controller and the transmitter are in a stable position and orientation. The newly calculated mesh is averaged with the existing mesh data, which helps reduce mesh error due to unintended or rapid movement.

The second rule states that we only update the part of the mesh that is both visible in the current depth frame and lies within the interaction space. The third rule states that, if a real object is positioned in front of the mesh causing an occlusion, then the mesh behind the real object will not be updated. Due to the

single point of view, the depth camera is blind to the space behind occluding objects, and we assume that the mesh behind the object stays the same.

3.2.1.3 System Usage

Initialization of the system is simple with only two steps. First, the user must define the size of the interaction space, which is the area that the system will use for environment mapping. This allows the user to choose optimal settings for different game scenarios, environments, and hardware capabilities.

Next, the user places the Razer Hydra transmitter in the environment, and the interaction space is centered on this point. The transmitter can be moved around freely at any time as the mesh is only updated when certain conditions are met, as described in mesh reconstruction.

3.2.2 Modalities of Hand-based Interaction

KITE offers four modalities for hand-based interaction. They are touch screen, controller, behind and in-front as shown in Figure 3.13. Touch screen and controller interaction offer tactile feedback and are therefore most appropriate for interaction that requires precision. Behind the screen hand interaction (Piumsomboon et al., 2012), and in-front gestures are both captured using the hand tracking functionality in OpenNI, and require the depth camera to be pointed away from and toward the user respectively. Ideally, two depth cameras would be used for simultaneous in-front and behind interaction, however this is currently not possible due to the limited bandwidth of the USB controller. Unique functionalities in each modality are menu selection for touch screen, direct control with the controller, direct physical manipulation on the virtual

content with the hand behind the KITE, and hand gesture with the hand in front of the KITE.

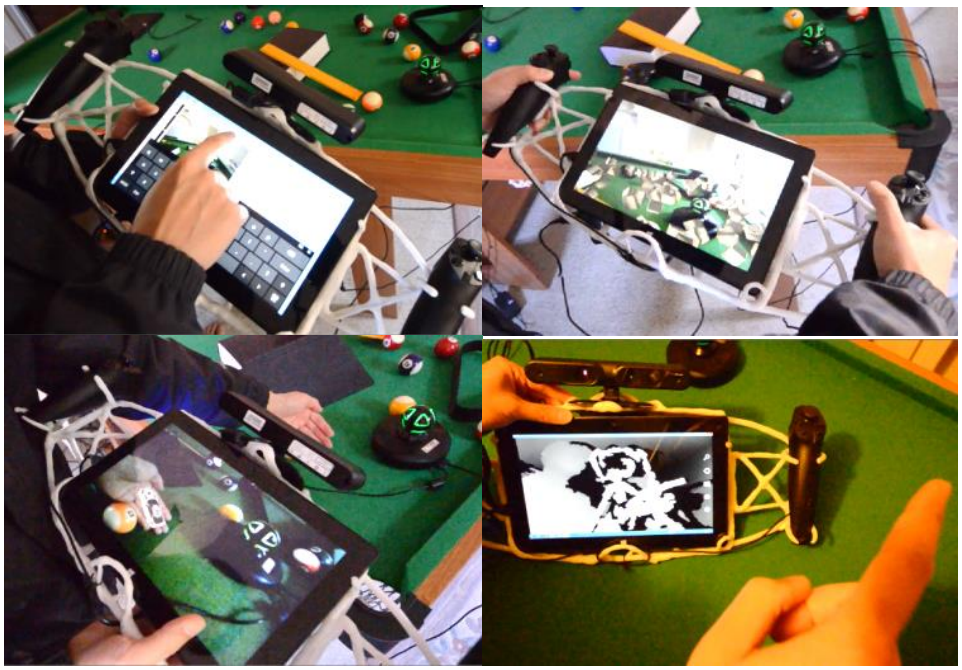


Figure 3.13: KITE offers four modalities for hand-based interaction, which are touch (top-left), controller (top-right), behind (bottom-left) and in-front (bottom-right)

3.2.3 Performance

The Razer Hydra tracker removes the computational requirements of vision-based tracking, freeing up processor and memory usage for simulation and interaction. The tracking error is below 1mm in position and 1 degree in orientation when the Razer Hydra controller is within 1.5 m from its transmitter. The standard deviation of error for position and orientation are graphed in Figure 3.14.

The Primesense Carmine 1.09 has an optimal operational range between 0.35-1.40 m from the surface scanned. On a Microsoft Surface Pro with an Intel Core i5 processor and 4GB RAM, a depth resolution of 320x240 pixels yields a framerate of 10-15 fps, while a resolution of 640x480 pixels reduces the frame rate to below 10 fps.

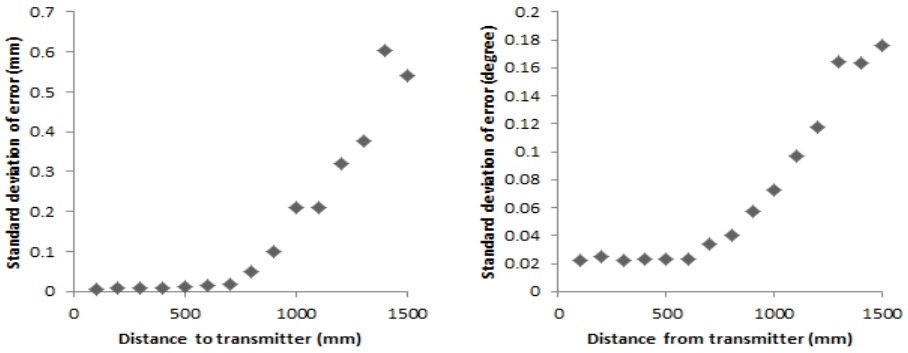


Figure 3.14: Graphs show standard deviation of error and distance between the Razer Hydra controller to its transmitter for position (left) and orientation (right)

3.2.4 Demonstrations

The first demonstration, a rigid-body dynamic simulation, is a stress test program that drops hundreds of virtual boxes into the real scene, with physical interaction between the mesh that represents real objects and the virtual boxes simulated. The second demonstration is a car racing game, where the user controls the virtual car with the Razer Hydra controllers. Real objects in the scene can be used by players to create their own custom obstacle courses.

Both demonstrations have been shown to a small group of students, with early feedback showing an overall positive impression. From the initial feedback, a number of possible improvements have been identified, for example, holding the KITE for a long period can cause fatigue, and providing a strap that distributes the weight on to the user's shoulders could alleviate such

discomfort. Section 3.3 discusses further findings as a generalization from demonstrations of both systems.

3.3 Discussion

From feedback obtained during public demonstrations of both systems, users found physically-based interaction and environment awareness to be a natural and intuitive way to interact with virtual contents in AR. In this section, we discuss observations made on user interaction during the demonstrations in Section 3.3.1, the limitations of the two systems are discussed in Section 3.3.2, and we conclude with the characteristics of affordance found in Section 3.3.3

3.3.1 Observations and Feedback

Users found interacting with both systems to be natural and intuitive. With environment awareness, virtual objects were placed correctly in the physical environment with appropriate occlusion and shadows, which increased the perception that they were real. In turn, users could use physical objects to physically interact with virtual objects. Although only simple physical actions such as pushing, lifting etc. could be performed directly on the virtual objects, users found these actions satisfying for the purposes of the demonstrated application. The interaction was intuitive and easy for users to understand. Users could learn the actions by observing other users and start interacting without any further explanation.

Many users also enjoyed exploring the limit of the interaction by trying new actions and introducing new objects into the scene. Some users felt that if the virtual contents afforded a physical manipulation, then they should also behave and react like a real world objects. For example some users attempted actions such as squashing the virtual objects with their fist.

3.3.2 Limitations

Although the feedback was overwhelmingly positive, several issues were raised related to limitations of the underlying technology.

3.3.2.1 Single Viewpoint Depth Sensing

As only a single depth sensor is used in both the tabletop and mobile system, there is only one viewpoint of the scene, resulting in blind spots. This can cause discrepancies between the real and virtual world if there are physical changes happening in those area. Additionally, any physical characteristics that are not visible to the sensor will not be accurately modeled, for example if a cup is placed on its side in the scene facing away from the camera, the system would not be aware that the cup is hollow. To solve this issue, multiple sensors should be used to provide multiple viewpoints and cover potential blind spots. In the case where multiple users are using mobile setups with depth sensors, a unique view from each user can be used to model the environment more accurately.

3.3.2.2 Level of Precision of Physically-based Interaction

The level of precision that can be achieved using the physically-based interaction was also an issue. Current hardware limitations create problems during detection of interaction in regards to both space and time. The spatial issue is due to the resolution of the depth image and the distance of the camera to the targeted object. The temporal issue is due to the camera's framerate, limiting the update frequency of the simulation and the speed of camera and object movement in the scene.

To further elaborate on the spatial issue, when approximating the hand using a mesh or spheres the precision depends on the nature of proxy particles. The structure of the proxy particles created depends on the detected skin pixels, and

as the system does not have an understanding of the complete 3D structure of the hand. Consequently, there were only a limited number of gestures that users could perform, and common gestures such as pinching an object in the sensor's blind spot were not possible.

In the demonstrations, to maximize the performance of the simulation, the scene was updated at 10 Hz and the depth image's resolution was limited to 160 by 120 pixels. Furthermore, although smoothing was applied to improve the depth image, some noise was still present which reduced the quality of the reconstruction. Higher processing capabilities and better quality sensors would permit a higher update rate and lower noise level.

Another limitation with the current system is a lack of tactile and haptic feedback. Physically-based interaction through natural hands benefits from tactile or haptic feedback to improve the precision of the interaction. Although it is not the focus of this research to explore with haptic technology, visual and audio feedback could be used to improve the overall experience.

3.3.2.3 Display Setups

In the first system, a screen fixed in place in the environment was used for showing the viewpoint from the camera. The user could hold the camera and move it around or leave it fixed in place, however, the user still needed to look away from the interaction space to view the screen. This disconnect between the display and the interaction space could hinder the user experience and reduce the level of immersion.

In the second system, we used a tablet with camera attached such that the view shown on the screen acted as if the user was looking through a window into the real world. The display and interaction space were more connected this way, however, whenever the user wanted to reach out to directly manipulate the

object, they had to move their arms around the tablet. Furthermore, as one hand was needed to hold onto the device, bimanual operation was impossible.

3.3.3 Characteristics of Affordance

The characteristics of affordance found for AR systems that offer environment awareness, physically-based interaction, and simple natural hand interaction are as follows:

Physical actions are inherent to interaction in AR. As AR overlays virtual 3D content into the real environment, the user's expectation of interactivity matches that of the physical world. For this reason, tangible interfaces are a natural choice for interaction in AR. For this reason, people feel that it is more natural and intuitive to interact with systems that offer environment awareness and physics-enabled simulation.

Virtual appearance could be associated with a real world affordance. When users see a virtual content that resembles a real world object, they anticipate that the virtual object offers the same affordances as the real one.

3.4 Lessons Learnt and Research Questions Raised

The lessons learnt from this chapter and important research questions raised are summarized in this section. These steer the research direction of this thesis and will be covered in the chapters to follow.

3.4.1 Physically-based Natural Hand Interaction is Natural but Restricted

Natural hand interaction supports direct manipulation of the virtual objects through physical simulation. Although it is natural and intuitive, it is very limited in term of functionality. There are other types of interaction that use hands as a medium which weren't explored, such as the use of hand gestures.

Would users benefit from being able to use hand gestures in an AR environment? Are there enough unique gestures to map to all the desired commands? How can we assign gestures to appropriate commands so that they remain intuitive to use?

Furthermore, as explained in Section 3.3.2.2, the current technique that substitutes skin color pixels with proxy particles is limited in term of precision. Another technique must be used to improve the precision.

We explores these questions in Chapter 4: User-defined Gestures in AR and Chapter 5: Development of Gesture Interface for Augmented Reality.

3.4.2 Common Components in Software Architecture for AR

In this chapter, two AR systems were developed. Both systems make use of a depth sensor where one has the sensor fixed in the environment and the other is mobile. Both systems have a similar architecture and common components, which are; (1) scene tracking to determine the transformation of the viewing camera, (2) scene reconstruction to map the environment, (3) scene understanding to recognize and track objects in the scene (4) physics simulation to enable collision detection and dynamics, and (5) rendering to create visual, audio and other feedback. These components are fundamental to the creation of AR systems, and we reuse these components to create an AR framework that supports multimodal inputs in Chapter 6: Multimodal Augmented Reality Framework and Interaction Techniques.

3.5 Conclusion

In this chapter, two AR systems were presented. They employed a depth sensor to explore environment awareness and physically-based interaction in AR. The first system was designed for face-to-face collaboration on a tabletop setting,

while the second was created as a platform for mobile gaming that utilized a magnetic tracker instead of an image-based marker. The depth sensor provided depth information in real-time, which enabled scene understanding and provided seamless interaction between the real and virtual objects.

In the tabletop-based collaborative system, natural hand interaction allowed for direct manipulation of virtual contents in AR. The methods developed for hand segmentation and geometric representation of the hand, including mesh-based and sphere substitution, were discussed. In the second system, four modalities of hand inputs were proposed including touch input on the computing surface, control with the controller, direct hand interaction with virtual content using the rear depth sensor, and gestures input using the front sensor.

Numerous demonstrations were given to the public, and observations and feedback were collected. It was found that direct physical interaction using hands and physical objects through environment awareness and physically-based interaction was natural and intuitive. This led to a conclusion that physical actions are inherent to interaction in AR.

Despite its merits, the direct natural hand interaction technique presented in this chapter is limited to a basic physical manipulation. A common method of supporting more functionality while still keeping natural hand interaction natural and intuitive is through the use of gestures, however this is an area that has not been well studied from a user experience point of view. In the next chapter we present our guessability study, with the aim of supporting natural hand gesture input in the most natural and intuitive way possible.

Chapter 4
User-defined Gestures in Augmented Reality

In Chapter 3, two systems were created to explore and better understand environment awareness and physically-based interaction in AR. It was found that interaction through direct manipulation using natural hands can be natural and intuitive, however, it is very limited in term of functionality. A common method of improving the functionality of a natural hand interface while ensuring it remains natural and intuitive is through the use of hand gestures. To determine whether gestures are effective and how best to support gestural interaction in AR, there are a number of questions which need to be answered: Would users benefit from being able to use hand gestures in an AR? Are there enough unique gestures to map to all the desired commands? How can we assign gestures to appropriate commands so that they remain intuitive to use? In particular, this last point has often been overlooked in previous research, despite being critical to user experience.

In this chapter, Section 4.1 describes the problem in greater detail and outlines the contributions of this study. Section 4.2 explains the experiment in detail, while the experimental result is covered in Section 4.3. We discuss the implications and limitations of this study in Section 4.4. The lessons learnt from this study are summarized in Section 4.5. Lastly, we conclude the chapter in Section 4.6.

4.1 Introduction

While prior research has demonstrated the use of hand gestures in AR, there is no consensus on how this interaction technique can best serve users. In studies involving multimodal AR interfaces, hand gestures were primarily implemented

as an add-on to speech input (Heidemann et al., 2004; Mathias Kolsch et al., 2006). In cases of unimodal gesture interfaces, only a limited number of gestures have been used and the gestures were designed by researchers for optimal recognition rather than for naturalness, meaning that they were often arbitrary and unintuitive (Fernandes & Fernandez, 2009; Kaiser et al., 2003; T. Lee & Hollerer, 2007). Recent research, including our work presented in Chapter 3, has integrated hand tracking with physical simulation to provide realistic interaction with virtual content (Benko et al., 2012; Hilliges et al., 2012), but this technique can only provide restricted support for gesture recognition. Furthermore, it does not take into account the wide range of expressive hand gestures that could potentially be used for input.

To develop truly natural hand and gesture interaction based interfaces for AR, there are a number of unanswered questions that must be addressed. For example, for a given task is there a suitable and easy to perform gesture? Is there a common set of gestures among users that would eliminate the need for arbitrary mapping of commands by designers? Is there a taxonomy that can be used to classify gestures in AR? Similar questions were encountered in the areas of uni-stroke gestures (Wobbrock, Aung, Rothrock, & Myers, 2005), surface computing (Wobbrock, Morris, & Wilson, 2009) and motion gestures (Ruiz, Li, & Lank, 2011), where researchers addressed the absences of design insight by conducting guessability studies (Wobbrock et al., 2005). This technique is a common method to gain insights into design practice from the user's perspective in participatory design (Schuler & Namioka, 1993).

In this study, the focus is on hand gestures for unimodal input in AR. Wobbrock's approach has been adopted and a guessability method is employed, by first showing a 3D animation of the task we wish to define a gesture for, and then asking the user for their preferred gesture to perform the task. Users were also asked to subjectively rate their chosen gestures, based on the perceived

"goodness" and ease to perform. The analysis of the results yielded a comprehensive set of user-defined gestures for a range of tasks performed in AR.

This study makes a number of contributions; (1) The first set of user-defined gestures captured for an AR interface, (2) Classification of these gestures based on a gesture taxonomy for AR which was extended from Wobbrock's surface gesture taxonomy (Wobbrock et al., 2009), (3) Agreement scores of gestures for selected tasks and subjective rating of the gestures, (4) Qualitative findings from the design process, and (5) Discussion of the implications of this work for AR, gesture interfaces, and gesture recognition.

4.2 Developing a User-defined Gesture Set

A guessability study was conducted in order to elicit gestures from users. The study began by showing each participant a task being carried out as a 3D animation in AR, with the participant then asked to describe the gestures they would use to perform this task. Participants designed and performed gestures for forty tasks across six categories, including gestures for three types of menu. Participants were asked to follow a think-aloud protocol while designing the gestures, and also to rate the gestures for "goodness" and "ease to perform". They were asked to ignore the issue of recognition difficulty to allow freedom during the design process, and to allow us to observe their unrestricted behavior. At the end of the experiment, brief interviews were conducted, and the participants were also asked to rank their preference for the three types of proposed gesture menus.

4.2.1 Task Selections

In order for the gesture set to be applicable across a broad range of AR applications (van Krevelen & Poelman, 2010), common tasks in previous research were examined (Broll et al., 2004; Fernandes & Fernandez, 2009; Kolsch et al., 2006; Lee et al., 2010; Lee & Hollerer, 2007), resulting in forty tasks that included three types of gesture menu: horizontal (Lee et al., 2004), vertical (Broll et al., 2004), and an object-centric menu proposed for this study. These tasks were grouped into six categories based on context, as shown in Table 4.1, such that the same gestures could be reused in different categories.

4.2.2 Participants

Twenty participants were recruited for the study, comprising of twelve males and eight females, ranging in age from 18 to 38 with mean age of 26 ($\sigma = 5.23$). The participants selected had minimal knowledge of AR to reduce any influence of previous experience with gesture interaction in AR. Nineteen of the participants were right handed, and one was left handed. All participants used PCs regularly, with an average daily usage of 7.25 hours ($\sigma = 4.0$). Fifteen owned touchscreen devices, with an average daily usage of 3.6 hours ($\sigma = 4.17$). Eleven had experience with gesture-in-the-air interfaces such as those used by the Nintendo Wii or Microsoft Kinect gaming devices.

Table 4.1: The list of forty tasks in six categories.

Category		Tasks	Category		Tasks
Transforms	Move	1. Short distance	Editing		21. Insert
		2. Long distance			22. Delete
		3. Roll (X-axis)			23. Undo
	Rotate	4. Pitch (Y-axis)			24. Redo
		5. Yaw (Z-axis)			25. Group
		6. Uniform scale			26.Ungroup
	Scale	7. X-axis			27. Accept
		8. Y-axis			28. Reject
		9. Z-axis			29. Copy
Simulation		10. Play/Resume			30. Cut
		11. Pause			31. Paste
		12. Stop/Reset	Menu	Horizontal	32. Open
		13.Increase speed		(HM)	33. Close
		14.Decrease speed			34. Select
Browsing		15. Previous		Vertical	35. Open
		16. Next		(VM)	36. Close
Selection		17.Single selection			37. Select
		18.Multiple		Object-	38. Open
		selection		centric	
		19.Box selection		(OM)	39. Close
		20.All selection			40. Select

4.2.3 Apparatus

The experimental interaction space, shown in Figure 4.1 (Left), was the area on and above a 120 x 80cm table. Each participant was seated in front of the table, and a Sony HMZ-T1 head mounted display (HMD) with a resolution of 1280 x 720 pixels was used as the display device. A high definition (HD) Logitech c525 web camera was mounted on the front of the HMZ-T1 as a viewing camera, providing a video stream at the display resolution. This HMD and camera combination offered a wide field of view, with a 16:9 aspect ratio, providing a good view of the interaction space and complete sight of both hands while gesturing.

An Asus Xtion Pro Live depth sensor was placed 100 cm above the tabletop facing down onto the surface to provide reconstruction and occlusion between the user's hands and virtual content. An RGB camera was placed in front of the user and facing them to record the users' gestures. A PC was used for the AR simulation and to record the video and audio stream from the user's perspective. A planar image marker was placed in the center of the table, and the OPIRA natural feature registration library (Clark et al., 2008) was used for registration

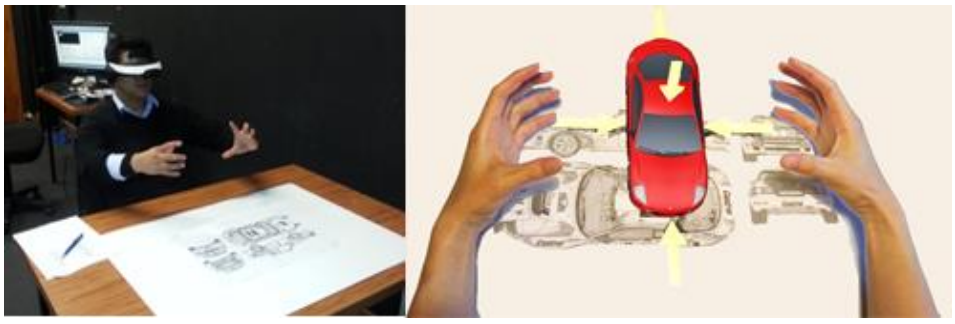


Figure 4.1: (Left) A participant performs a gesture in front of the image marker. (Right) The participant sees an AR animation of a shrinking car, and performs their gesture for a uniform scale task

and tracking of this marker. The 3D graphics, animation and occlusion techniques were implemented from the work by Piumsomboon et al. (2012).

4.2.4 Procedure

After an introduction to AR and description of how to operate the interface, the researcher described the experiment in detail and showed the list of tasks to the participant. The forty tasks were divided into six categories, as shown in Table 4.1, and the participant was told they could choose to carry out the categories in any order, providing that there was no conflict between gestures within the same category. For each task, a 3D animation showing the effect of the task was displayed, for example, in the "Move – long distance" task, participants would see a virtual toy block move across the table. Within the same task category, the participant could view each task as many times as she/he needed. Once the participant understood the function of the task, she/he was asked to design the gesture they felt best suited the task in a think-aloud manner. Participants were free to perform one or two-handed gestures as they saw fit for the task (See Figure 4.1, Right).

Once the participant had designed a consistent set of gestures for all tasks within the same category, they were asked to perform each gesture three times. After performing each gesture, they were asked to rate the gesture on a 7-point Likert scale in term of goodness and ease of use. At the end of the experiment, a final interview was conducted, where participants were asked to rank the three types of menu presented (horizontal, vertical, and object-centric as shown in Figure 4.5) in terms of preference and the justification for their ranking. Each session took approximately one to one and a half hours to complete.

4.3 Result

A total of 800 gestures were generated from the 20 participants performing the 40 tasks. The data collected for each user included video and audio recorded from both the camera facing towards the user and the user's viewpoint camera, the user's subjective rating for each gesture, and transcripts taken from the think-aloud protocol and interview.

4.3.1 Taxonomy of Gestures in AR

The topic of hand gesture classification as based on human discourse was excellently covered by the work of Wobbrock et al. (Wobbrock et al., 2009). Wobbrock's surface taxonomy was adapted to better cover the AR gesture design space by taking their four-dimensional taxonomy, (*form*, *nature*, *binding*, and *flow)* and extending it with two more dimensions; *symmetry* and *locale*. Each dimension is comprised of multiple categories, as shown in Table 4.2.

The scope of the *form* dimension was kept unimanual, and in the case of a two-handed gesture, applied separately to each hand. In Wobbrock's original taxonomy, *form* contained six categories including *one-point touch* and *one-point path*, however, these two categories were discarded as they were not relevant to AR gestures that occur in three dimensional space.

The *nature* dimension was divided into *symbolic*, *physical*, *metaphorical* and *abstract* categories. Examples of symbolic gestures are thumbs-up and thumbs-down for *accept* and *reject*. *Physical* gestures were classified as those that would act physically on the virtual object as if it was a real object, for example grabbing a virtual block and relocating it for a *move* task. *Metaphorical* gestures express actions through existing metaphors e.g. pointing an index finger forward and

Table 4.2: Taxonomy of gestures in AR extended from surface gestures

Taxonomy of Gestures in AR		
Form	static pose	Hand pose is held in one location.
	dynamic pose	Hand pose changes in one location.
	static pose and path	Hand pose is held as hand relocates.
	dynamic pose and path	Hand pose changes as hand relocates.
Nature	Symbolic	Gesture visually depicts a symbol.
	physical	Gesture acts physically on objects.
	metaphorical	Gesture is metaphorical.
	abstract	Gesture mapping is arbitrary.
Binding	object-centric	Gesturing space is relative to the object.
	world-dependent	Gesturing space is relative to the physical world.
	world-independent	Gesture anywhere regardless of position in the world.
	mixed dependencies	Gesture involves multiple spaces.
Flow	Discrete	Response occurs after the gesture completion.
	continuous	Response occurs during the gesture.
Symmetry	dominant unimanual	Gesture performed by dominant hand.
	nondominant unimanual	Gesture performed by nondominant hand.
	symmetric bimanual	Gesture using both hands with the same form.
	asymmetric bimanual	Gesture using both hands with different form.
Locale	on-the-surface	Gesture involves a contact with real physical surface.
	in-the-air	Gesture occurs in the air with no physical contact.
	mixed locales	Gesture involves both locales.

spinning it clockwise to indicate *play* or *increase speed* as if one was playing a roll film. Any arbitrary gestures were considered *abstract*, such as a double-tap on the surface to *deselect* all objects.

The *binding* dimension considered the relative location where gestures were performed. The *object-centric* category covered *transform* tasks such as *move*, *rotate*, and *scale*, as these are defined with respect to the objects being manipulated. Opening and closing horizontal or vertical menus were classified in the *world-dependent* category as they are located relative to the physical workspace. Gestures in the *World-independent* category could be performed anywhere, regardless of the relative position to the world, such as an open hand facing away from one's body to indicate *stop* during a simulation. Gestures performed across multiple spaces, such as *insert* where selection is *object-centric* and placement is *world-dependent*, fell into the *mixed dependencies* category.

In the *flow* dimension, gestures were categorized as *discrete* when the action is taken only after the gesture is completed, for example an index finger must be spun clockwise in a full circle to perform the *play* command. The gestures were considered *continuous* if the simulation must respond during the operation, such as manipulating an object using the *transform* gestures.

The first additional dimension proposed in this work, *symmetry*, allowed classification of gestures depending on whether they were one handed (*unimanual*) or two handed (*bimanual*). The *unimanual* category was further split into *dominant* and *non-dominant*, as some participants preferred to use their non-dominant hand to perform gestures that required little or no movement, leaving their dominant hand for gestures requiring finer motor control. An example of this would be to use the dominant hand to execute a *selection*, and then use the non-dominant hand to perform a scissor pose for a *cut* operation. The *bimanual* category was also subdivided into *symmetric* and *asymmetric* gestures. *Symmetric* gestures represented two-handed gestures where both hands executed the same *form*, for example scaling, where both hands perform a pinch moving toward or away from each other. Two handed gestures, where

the *form*s of the hands are different, fall into the *asymmetric bimanual* category. An example of this is the *copy (1)* gesture where one hand is used to select the target object (*static pose*) while the other hand drags the copy into place (*static pose and path*).

The other dimension introduced was *locale*. If a gesture required physical contact with the real surface, it is considered *on-the-surface* as opposed to *in-the-air*. Gestures that require both are considered *mixed locales*. For example, an index finger tap on a virtual button projected on the tabletop surface would be an *on-the-surface* gesture, as opposed to an index finger pushed into a floating button, which is an *in-the-air* gesture. An example of a *mixed locales* gesture is *box selection (1),* where one hand indicates the area of the bottom surface of the box by dragging an index finger diagonally along the table's surface, while another hand lifts off the surface into the air to indicate the height of the box (See Figure 4.5).

4.3.2 Findings from Classification

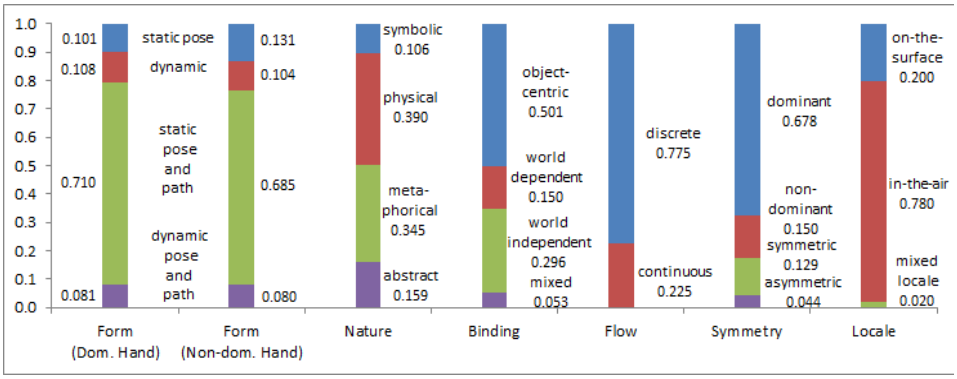


Figure 4.2: The proportion of gestures in each category in the six dimensional taxonomy. Form has been calculated for each hand separately

Classification was performed on the 800 gestures as shown in Figure 4.2. Within the six dimensional taxonomy, the most common characteristics of gestures

were *static pose and path*, *physical*, *object-centric*, *discrete*, *dominant unimanual*, and *in-the-air*.

Within the *form* dimension, there was a slightly higher number of *static poses* (3%) performed with a *non-dominant* hand and lower for *static poses with path* gestures (2.5%) over a *dominant* hand. This slight discrepancy was caused by some participants preferring to use their *dominant* hand for gestures with movement while using their *non-dominant* hands for static poses.

In the *nature* dimension, overall the gestures were predominantly *physical* (39%) and *metaphorical* (34.5%). The gestures chosen to perform *transform*, *selection*, and *menu* tasks were predominantly *physical*, with the percentage of 76.1%, 50%, and 57.8% respectively. The *browsing* and *editing* task gestures were mainly *metaphorical* (100% and 40.9% respectively), while the *simulation* task, gestures were split across *symbolic* (37%), *metaphorical* (34%), and *abstract* (29%) categories. For the *binding* dimension, the majority of gestures for the *transform* and *selection* tasks were *object-centric* (100% and 75% respectively). *Simulation* (93%) and *browsing* (100%) task gestures were mainly *world-independent* (93% and 100%), while *editing* tasks gestures were *world-independent* (39.5%) and *object-centric* (32.3%). *Menu* tasks gestures were *object-centric* (50%) and *world-dependent* (45.6%).

For the remaining dimensions including *flow*, *symmetry*, and *locale*, the gestures chosen across all tasks were primarily *discrete* (77.5%), *dominant unimanual* (67.8%) and *in-the-air* (78%).

4.3.3 A User-defined Gesture Set

As defined in prior work by Wobbrock et al. (Wobbrock et al., 2009) and Ruiz et al. (Ruiz et al., 2011), the user defined gesture set, known as the "consensus set", is constructed based on the largest groups of identical gestures that are performed for the given task. In our study, each gesture was given a value of

one point; therefore there were 20 points within each task and a total of 800 points for all tasks.

It was found that most participants used minor variations of similar hand poses, for example a swiping gesture with the index finger or the same swipe with the index and middle fingers, and we therefore decided to loosen the constraints from "gestures must be identical within each group" to "gestures must be similar within each group". "Similar gestures" were defined as *static pose and path* gestures that were identical or having consistent directionality, even though the gesture had been performed with different *static* hand poses.

The major variants of observed hand poses had been classified into 11 poses with the codes, *H01* to *H11*, as illustrated in Figure 4.4. For tasks where these variants existed, the variant poses could be used interchangeably, as indicated by the description under each user-defined gesture's illustration (Figure 4.5).

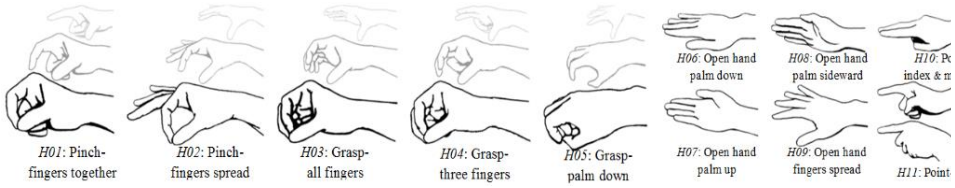


Figure 4.3: Variants of hand poses observed among gestures where the codes, H01-H11, were assigned for ease of reference

By applying the "similar gesture" constraint, the original 800 gestures were reduced into 320 unique gestures. The top 44 most highly scored gestures were selected to make the consensus set, while the remaining 276 lowest scored gestures were discarded, defined by Wobbrock et al. (Wobbrock et al., 2009) as the discarded set. The selected gestures of the consensus set represented 495 (61.89%) of the 800 recorded gestures (495 of 800 points). The consensus set of gestures comprised the overall task gestures in the following percentage *transform* (19.38%), *menu* (17.75%), *editing* (11.75%), *browsing* (5.00%), *selection* (4.63%), and *simulation* (3.38%), with a total sum of 61.89%.

4.3.2.1 Level of Agreement

To compute the degree of consensus among the designed gestures, an agreement score *A* was calculated using Equation 4.1 (Wobbrock et al., 2005):

$$A = \sum_{P_s} \left(\frac{|P_s|}{|P_t|} \right)^2 \quad (4.1)$$

Where P_t is the total number of gestures within the task, *t*, P_s is a subset of P_t containing similar gestures, and the range of *A* is [0, 1].

Consider the *rotate-pitch* (*y-axis*) task that contained five gestures with scores of 8, 6, 4, 1, and 1 points. The calculation for A_{pitch} is as follows:

$$A_{pitch} = \left(\frac{|8|}{|20|} \right)^2 + \left(\frac{|6|}{|20|} \right)^2 + \left(\frac{|4|}{|20|} \right)^2 + \left(\frac{|1|}{|20|} \right)^2 + \left(\frac{|1|}{|20|} \right)^2 = 0.295 \quad (4.2)$$

The agreement scores for all forty tasks are shown in Figure 4.3. While there is low agreement in the gestures set for tasks such as *all select*, *undo*, *redo* and *play*, there were notable groups of gestures that stood out with higher scores.

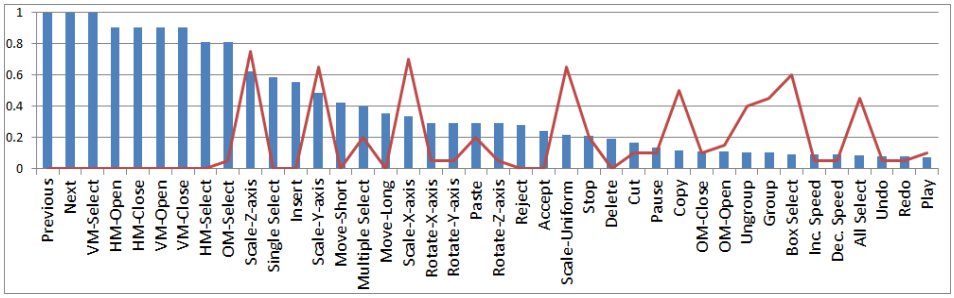


Figure 4.4: Agreement scores for forty tasks in descending order (bars) and ratio of two-handed gestures elicited in each task (line)

4.3.2.2 User-defined Gesture Set and Its Characteristics

As mentioned in Section 4.2.4, users were allowed to assign the same gesture to different tasks as long as the tasks were not in the same category. In addition to this, there were some tasks where there were two or more gestures commonly assigned by the participants. This non "one-to-one" mapping between gestures and tasks resulted in a consensus set of 44 gestures for a total of 40 tasks, which resulted in improved guessability (Wobbrock et al., 2005).

For single tasks represented by multiple gestures, there was one task that had three gestures assigned to it (*uniform-scaling*), and seven tasks that had two gestures (*x, y, z scaling, box select, stop, delete, and copy*). For single gestures representing multiple tasks, two gestures were assigned to four tasks (*short, long move, insert, and paste*), one gesture assigned to three tasks (*play, increase speed, and redo*), and one gesture assigned to two tasks (*decrease speed and undo*). The remaining tasks and gestures had a one-to-one mapping.

When creating the consensus set, one conflict was found between gestures within the same category. This was between the *pause* and *stop* tasks, where the gesture of an open-hand facing away from the body was proposed for both with scores of 4 and 7 points respectively. To resolve this, the gesture was simply assigned to the task with the higher score, in this case *stop*.

Play and *increase speed* as well as *insert* and *paste* were exceptions where a single gesture was assigned to two tasks within the same category with no conflict. For *play* and *increase speed*, the participants intention was to use the number of spin cycles of the index finger to indicate the speed of the simulation i.e. a single clockwise spin to indicate *play*, two clockwise spin to indicate *twice* the speed and three spins for *quadruple* speed. For *insert* and *paste*, the participants felt the two tasks served a similar purpose; *insert* allowed a user to select the object from menu and placed it in the scene, whereas *paste* allowed a

user to place an object from the clipboard into the scene. In the follow up interviews, participants suggested a simple resolution to this would be to provide unique selection spaces for the *insert* menu and *paste* clipboard.

With these minor ambiguities resolved, a consistent set of user-defined gestures was constructed. It contained 44 gestures, where 34 gestures were unimanual and 10 were bimanual. The complete gesture set is illustrated in Figure 4.5.

4.3.2.3 The Subjective Rating on Goodness and Ease

After the participants had finished designing gestures for a task category, they were asked to subjectively rate their gestures for goodness and ease to perform on a 7-point Likert scale. By comparing these subjective ratings between the consensus set (user-defined set) and the discarded set, the average score for gestures that users believed were a good match for the tasks was 6.02 ($\sigma = 1.00$) for the consensus set and 5.50 ($\sigma = 1.22$) for the discarded set, while the average score for the ease of performance was 6.17 ($\sigma = 1.03$) for the consensus set and 5.83 ($\sigma = 1.21$) for the discarded set. The consensus set was rated significantly higher than the discarded set for both goodness ($F_{1, 798} = 43.896$, $p < .0001$) and ease of performance ($F_{1, 798} = 18.132$, $p < .0001$). Hence, it could be concluded that, on average, gestures in the consensus set were better than those in the discarded set in terms of goodness and ease of performance.

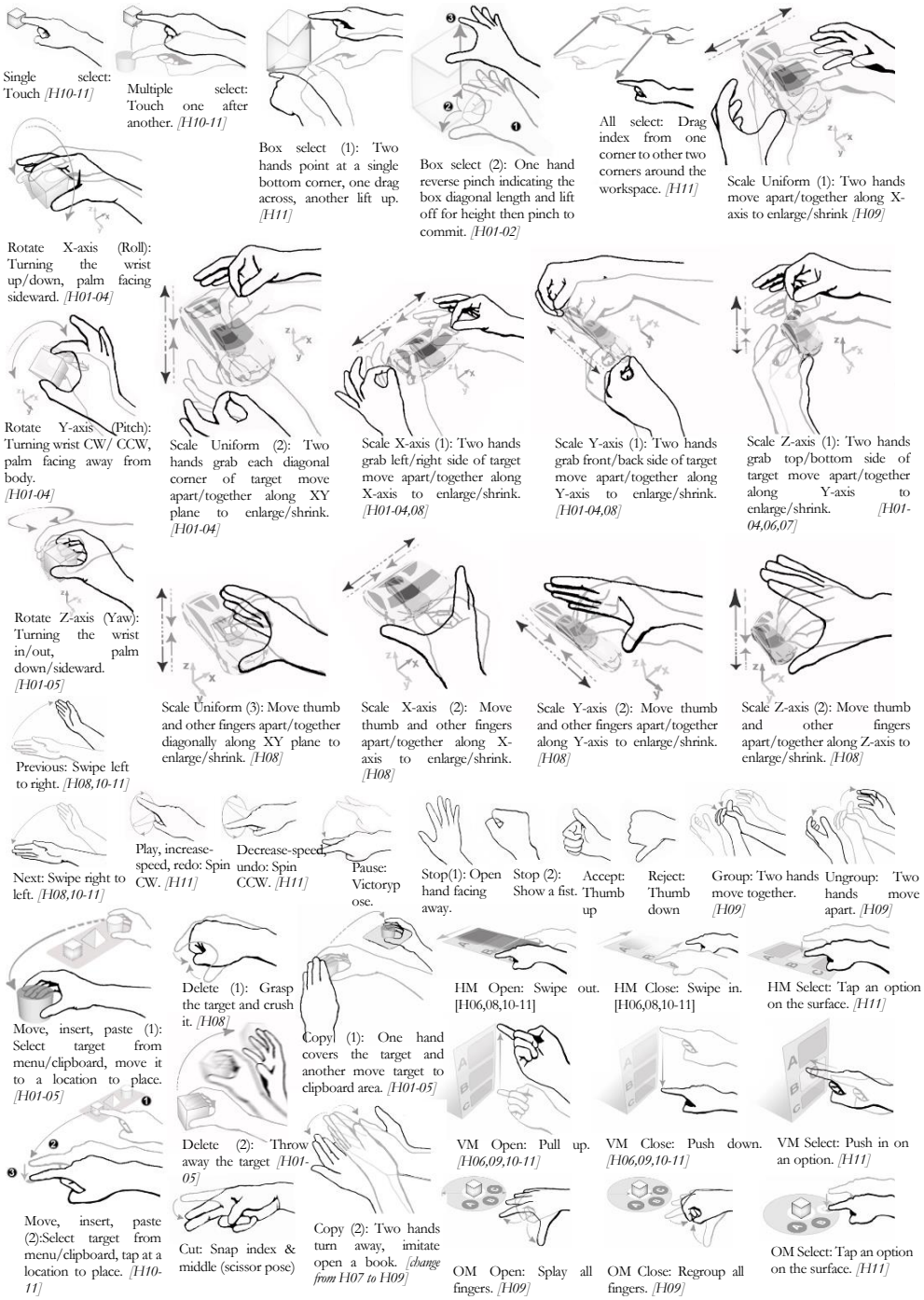


Figure 4.5: The user-defined gesture set for AR. The number shown in the parenthesis indicates multiple gestures in the same task. The codes in the square bracket indicate the hand pose variants (Figure 4) that can be used for the same gesture

4.3.4 Findings from the Design Process

Participants were asked to think-aloud when designing their gestures, and a follow-up interview was conducted after the experiment was complete. Analysis of the resulting empirical data showed recurring thought processes. We present seven motifs, which describe the common design patterns encountered in designing gestures for AR: *reversible and reusable, size does matter, influence from existing UI, the obvious and the obscure, feedback backfired, menu for AR, axes and boxes,* and *variation of hand poses*.

4.3.4.1 Reversible and Reusable

The consensus set included reversible and reusable gestures. Reversible gestures are defined as those when performed in an opposite direction yielded opposite effects e.g. *rotation, scaling, increase/decrease speed* etc. Reusable gestures are defined as those which were used commonly for tasks which were different, but participants felt had common attributes e.g. *increase speed/ redo*, *decrease speed/undo,* and *insert/paste*. In the experiment there were several dichotomous tasks, which are defined as individual tasks that perform the exact opposite operation. Participants used reversible gestures for tasks where the opposite effect was presented in a single animation, such as rotation and scaling, as well as tasks where the opposite effects were shown in a separate animation, such as *increase/decrease speed*, *previous/next*, *undo/redo*, *group/ungroup*, and *open/close menus*. All two-handed dichotomous tasks were symmetric bimanual with the gestures performed on both hands being the same form.

4.3.4.2 Size Does Matter

The virtual object's size was found to influence the design decision of some participants, especially with regards to the number of hands that they would use

to manipulate the object. For example the majority of gestures performed for *scale* tasks were bimanual, as scaling involving shrinking and enlarging the target object within and beyond the palm size. Some comments are as follows:

> *"Instinctively, I would use two hands to adapt to the size of the model but it's cool if I can use just the two fingers (one-handed) for something as large."* – P04

> *"Depending on the size of the piece, I can use two hands when it's big but in the case of small piece, it's enough to use the two fingers (thumb and index)."* – P12

4.3.4.3 Influence from Existing UI

When participants found it difficult to come up with a gesture for a particular task, they would often resort to using metaphors from familiar UI. For example when designing a gesture for the *delete* task, several participants imagined having a recycle bin that they could move the target object to. For other arbitrary tasks, users would often resort to *double-tapping*. Some examples of how participants explained these actions were:

> *"I would select and double-click… I'm thinking too much like Microsoft. It's just the thing that I'm used to."* – P10

> *"The way I do it on my phone is that I would scale like this and then tap it once."* – P14

4.3.4.4 The Obvious and the Obscure.

Gesturing in 3D space allows for higher expressiveness, which in turn led to use of gestures commonly used in the real-world. For example, there was a high level of agreement on the *symbolic* gestures thumbs up/down for *accept*/*reject*

with scores of 9 and 10 respectively (out of 20). This was also the case for *metaphoric* gestures such as a scissor gesture for the *cut* task with the score of 7. User's liked the idea of using these gestures from the real world, resulting in higher than average goodness and ease scores, with averages of 6.87/6.75 (σ =.35/.71) for thumbs up, 6.5/6.5(σ =.71/.85) for thumbs down and 6.5/6.67(σ =.84/.82) for the scissor gesture.

The majority of participants found it challenging to come up with metaphors to design gestures for 3D tasks that they referred to as "abstract", such as *box selection*. In this task, users' had to design a gesture to define the width, depth and height of a 3D bounding box around target objects for selection. There was little agreement upon a common gesture, with a low agreement score of 0.095. In cases where the agreement score is below 0.1, further rigorous studies and usability tests are recommended to select the best gesture for the task. One participant expressed an opinion that was shared by many others:

> *"I don't think that it's unsuitable (the proposed gesture) but it's just very arbitrary and there is not a lot of intrinsic logic to it. If somebody told me that this is how to do it then I would figure it out but it's not obvious. It's just an arbitrary way of selecting a 3D area."* - P11

4.3.4.5 Feedback Backfired

Our experimental design included the use of a 3D camera to support hand occlusion, which gave users some concept of the relative position between the virtual content and their hands, however some participants found it to be obtrusive. One example of this criticism was as follows:

> *"Your hand gets in the way of the object so it can be hard to see how you're scaling it."* – P11

We present some ideas on how to improve this in Section 4.4.1.

4.3.4.6 Menus for AR

There was no significant difference in menu ranking. Some participants favored the horizontal menu because it was simple, familiar, easy to use/understand, supported on-the-surface gestures for touch sensing and did not interfere with virtual content. Others disliked the horizontal menu and felt it did not take advantage of 3D space with some options being further away and hard to reach.

The majority of participants found the vertical menu novel, and some found it to be appealing, easy to understand and that it made a good use of space as the distance to all options was evenly distributed. However, some found it harder to operate as they needed to lift their hands higher for options at the top if the buttons were arranged vertically.

Finally, some participants liked the object-centric menu because it was unique and object-specific so they knew exactly which object they were manipulating. However, some participants thought that it was unnatural and harder to operate in a crowded workspace. Furthermore, the open/close gestures for the object-centric menu were not as obvious as the horizontal and vertical menus, as indicated by the low agreement score of 0.11, compared to 0.905 for more traditional menu types.

4.3.4.7 Axes and Boxes

The *rotation* and *scaling* tasks, allowed for three possible coordinate systems, local, global, and user-centric, which corresponded to the *object-centric*, *world-dependent*, and *world-independent* categories in the *binding* dimension. In practice, the transformations were mostly *object-centric*; the participant would perform gestures based on the direction of the transformation presented on the

object. This was expected because people would naturally perform these tasks physically and adapted their bodies and gestures to suit the operation.

To perform a *rotation*, participants would grasp the object with at least two contact points and would move their hand or turn their wrist accordingly. For *scaling* on 3 axes, participants would grasp or use open-hands to align with the sides of object and increase or decrease the distance between their hands to enlarge or shrink the virtual object in the same direction as the transformation. *Uniform scaling* was less obvious, for example some participants preferred using open hands moving along a single axis in front of them, as shown in Figure 4.5 *uniform scale (1),* while the other preferred grasping the objects' opposing corners and pushing or pulling along the diagonal axis of the object as shown in Figure 4.5 *uniform scale (2).* Some user's expressed uncertainty about how to perform the task for a round object, and suggested that bounding volumes must be provided to manipulate these objects.

4.3.4.8 Variation of Hand Poses

Variants of a single hand pose were often used across multiple participants, and sometimes even by a single participant. Common hand poses were clustered into eleven poses, as shown in Figure 4.4. Multiple hand poses should be able to be used interchangeably for each gesture in a given task.

4.4 Discussion

In this section, the implications of the guessability study for the fields of AR, gesture interfaces, and gesture recognition are discussed.

4.4.1 Implications for Augmented Reality

While our experiment was conducted in a tabletop AR setting, the majority of the user-defined gestures are equally suitable to be performed in the air. Only four gestures were *on-the-surface*: *select all, open, close,* and *select horizontal menu,* while three were *mixed locale, box select (1), insert* and *paste (2).* This opens up our gesture set to other AR configurations, including wearable interfaces.

For our experiment, hand occlusion was implemented to provide users with a better understanding of the relative positions of their hands and the virtual content. However, it was found in some cases that this could hinder the user experience, especially when the virtual objects are smaller than the user's hand, causing the hands to occlude the object completely. We recommend that virtual objects are rendered as if the user's hands were translucent rather than opaque, or that occluded objects are rendered as outlines to provide some visual feedback of the objects' location.

As discussed in the *axes and boxes* motif, a clear indicator of axes and bounding boxes should be provided during object manipulation tasks. Due to an absence of haptic feedback, visual feedback should be provided to inform users of the contact points between their hands and the virtual objects.

4.4.2 Implications for Gesture Interfaces

It was found that most of the gestures elicited were *physical* (39%). Wobbrock et al. reached a similar outcome for surface gestures and suggested using a physical simulation for handling these gestures. This approach was implemented by Hilliges et al. (Hilliges et al., 2012) and Benko et al. (Benko et al., 2012), who introduced "physically-based interaction", however only basic manipulations were demonstrated, with limited precision and control over the

virtual contents. It was suggested that better control could be achieved by manipulation of the dynamical constraints imposed by the simulation. Many gestures can be detected using the collision detection component in the physics simulation for tasks such as object selection, scaling etc.

In the *size does matter* motif, it was described how object size influences the number of hands used for manipulation. Since the resulting user-defined gesture set contains both one-handed and two-handed gestures for tasks such as scaling, our recommendation is to take advantage of both one and two- handed gestures to provide different levels of control. For example, in scaling tasks, by combining a snap-to feature for different granularities, unimanual scaling could offer snap-to in millimeter steps and bimanual in centimeter steps, as users tended to use one hand for an object smaller than their palm size and two when it is larger.

As mentioned in *the obvious and the obscure* motif, care must be taken when choosing gestures for tasks with low agreement scores. It is also crucial to perform follow up studies to determine usability by comparing these gestures, designer-refined gestures, menu options and even alternative modalities in case of multimodal interface.

4.4.3 Implications for Gesture Recognition

High degree of freedom hand pose recognition is achievable, however it is computationally expensive. In the *variation of hand poses* motif, a limited number of common poses were found (Figure 4.4), reducing the search space. Furthermore, the majority of the resulting gestures were *static pose and path*, which are simpler to recognize than *dynamic pose and path* gestures. This could lead to more accurate and faster pose recognition while retaining the same level of flexibility and precision for users.

4.4.4 Limitations

While this study aimed to be a flexible as possible, there was a number of limitations that are addressed in the following sections.

4.4.4.1 Interaction Space was Within an Arm-reachable Distance

The interaction space used in this study was on and above the tabletop area in front of the subject. All interactions occurred within an arm-reachable distance for the subject. Generally, interaction spaces in AR can be of any size depending on the interface and targeted application, therefore this space can extend beyond the tabletop setting into a room size, street size or larger. Consequently, certain gestures that require direct contact with the virtual object might not be applicable when the object is at a distance beyond arms reach.

4.4.4.2 Disregard of Social and Cultural Aspects

Morgado (2014) proposed an interesting concept where gestures for command of systems can go beyond mimicry and non-kinesthetic. He recommended that cultural aspects of gestures be considered when designing gesture interfaces. The direct link between existing societal meanings and new meanings in interaction should improve learnability and memorability of the gestures. Furthermore, he pointed out that this might benefits users who are illiterate, including children, as textual cues are not required.

From our study, there were notable findings regarding possible social and cultural aspects of gestures elicited. For example, subjects with similar social or cultural backgrounds chose similar gestures for the same task, hinting at the appeal of these gestures as suggested by Morgado (2014). However, further study needs to be conducted with better control to confirm its relevance, and thus we choose not to draw conclusions from this.

4.5 Lessons Learnt and Research Questions Raised

In the previous section, implications are summarized for three areas including AR, gesture interfaces, and gesture recognition. In this section, these findings are rearticulated as lessons that will be applied in the next part of this thesis, Part III: Designing and Evaluating Natural Hand Interaction in Augmented Reality:

4.5.1 Hand Pose and Gestures Recognition are Crucial

The elicited gesture set and the variation of hand poses indicated that high degree of freedom hand pose and gesture recognition is needed for accurate recognition. Therefore, at minimum the gesture interface should recognize a subset of hand poses from the set of all variations. The majority of the elicited gestures were static pose and path, hence tracking of gestures is mostly only required at a high level i.e. hand's position and orientation, and it is not required down to the fingertip precision for most gestures. As a result, high precision instruments such as gloves are not necessary, however accurate tracking of the hand's position and orientation is still crucial to user experience. The development of gesture interface to support hand pose and gesture recognition is covered in Chapter 5: Development of a Gesture Interface for Augmented Reality.

4.5.2 Physical Gestures Urge Interaction through Contacts

It was found that most of the gestures elicited were physical, which correlates with the finding in Chapter 3 that physical actions are inherent to interaction in AR. Our work and past research has demonstrated the use of physical simulation to handle direct physical interaction through contact (Hilliges et al., 2012; Benko et al., 2012). However, limited functionality and precision can be

achieved with this technique. We demonstrate a novel technique called Grasp-Shell (G-Shell) that overcomes this limitation and support physical gestures in Chapter 6: Multimodal Augmented Reality Framework and Interaction Techniques.

4.5.3 Hand Occlusion Rendering and Enhanced Visual Feedback

We have learnt that correct occlusion of the hand and virtual content is crucial for users to comprehend the relative positions of these objects. However, when the size of the virtual content is smaller than the palm, it can be completely occluded, making it difficult for the user to know when and where they are making contact with the object in the absence of haptic or tactile feedback. We explore different methods of rendering to improve user experience in Chapter 6: Multimodal Augmented Reality Framework and Interaction Techniques.

4.6 Conclusion

This chapter presented the results of a guessability study, which aimed to address research questions raised in the previous chapter regarding direct natural hand interaction. The study yielded a comprehensive set of hand gestures and the first set of user-defined gestures in AR was presented. This gesture set was then categorized into an extended gesture taxonomy for AR. The agreement scores and subjective ratings for the user defined gesture set were presented. Through the use of the agreement score found among the elicited gestures, 44 user-defined gestures were selected as a "consensus set". Although gestures were found for all 40 tasks, agreement scores varied, suggesting that some gestures are more universally accepted than others.

The study yielded similar findings to an earlier study conducted on gestures for surface computing, where the majority of all gestures elicited were physical

gestures. These gestures are performed with physical contact on the virtual object. This result supports our finding in Chapter 3, which stated that physical interaction is inherent to AR. Past research employed physical simulation for physical gestures, we enhance this interaction and demonstrate a novel technique called Grasp-Shell (G-Shell) that can support a wide range of physical gestures in Chapter 6.

Moreover, we found that high degree of freedom hand pose and gesture recognition is necessary to support the elicited gesture set and the variation of hand poses. To be able to demonstrate and evaluate this gesture set, at least a subset of the elicited gesture set must be supported by our gesture interface. This is our motivation for the development of gesture interface to support hand pose and gesture recognition in the next chapter.

Finally, the insights from the qualitative findings in this study led to a number of implications and design recommendations that are used throughout the remaining chapters.

Part III
Designing and Evaluating Natural Hand Interaction in Augmented Reality

Chapter 5
Development of a Gesture Interface for Augmented Reality

In Chapter 3, it was found that physical simulation is a fundamental component of AR systems, with physical interaction through tangible objects and natural hands being a natural and intuitive way of interacting with virtual content. However, despite its merits, physical interaction can only support basic direct manipulation.

In Chapter 4, it was found that hands are capable of higher expressiveness through gestures. A guessability study was conducted and the results showed a consistent variation of hand poses in the gestures elicited indicating that only a limited number of hand poses are needed to be recognized to support natural interaction. Although this simplifies the problem, the hand poses elicited still vary considerably and high degree of freedom hand pose estimation is necessary for accurate recognition. At the time of this research, there was no platform that could offer these functionalities, motivating us to develop a unified gesture interface that can utilize a depth sensor to track, classify, and recognize hand input to support better natural hand interaction.

In this chapter, our approach in designing and implementing a new gesture interface is described. The aim for this interface is to use a single depth and color camera to track the hands, support natural hand interaction, and recognize gesture input. In this chapter, we present background research in Section 5.1, details of the gesture interface architecture in Section 5.2 and the Discussion and Conclusion in Sections 5.3 and 5.4, respectively.

5.1 Gesture Interface Architecture

When designing the architecture of a gesture interface, reusability and integration between components are crucial considerations. We have developed as architecture divided into five modules; hardware interface, segmentation/tracking, classification, modeling, and gesture recognition, as shown in Figure 5.1. Each module is described in more detail in the following sub-sections.

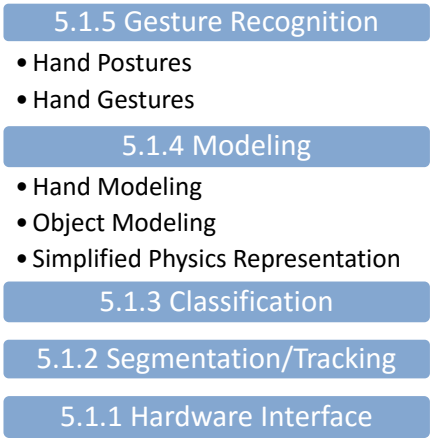


Figure 5.1: The overall gesture interface architecture

5.1.1 Hardware Interface

The hardware interface manages the input from depth sensors and converts this input into a common format so various hardware can be used interchangeably. It was implemented with modularity as a priority, and the hardware interface is extensible to support future hardware. It also supports simultaneous usage of multiple depth sensors from different manufacturers. To clarify the design of the hardware interface, this layer has been subdivided as shown in Figure 5.2.

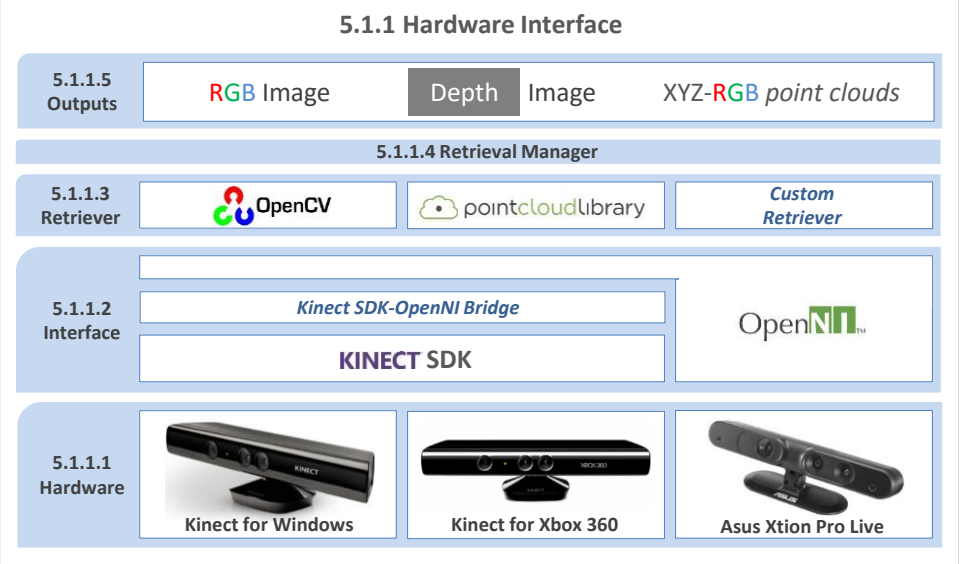


Figure 5.2: The hardware interface architecture

Table 5.1: Specifications of three consumer grade depth sensors

	Kinect for Windows	Kinect for Xbox 360	Asus Xtion Pro Live
Distance of Use	0.4-3.5 m (0.4-6 m)	1.2-3.5 m (0.7–6 m)	0.8-3.5 m (0.7-6 m)
Field of View	57° H, 43° V	57° H, 43° V	58° H, 45° V
Sensor	RGB & Depth	RGB & Depth	RGB & Depth
Depth Image Size	VGA (640x480) : 30 fps	VGA (640x480) : 30 fps	VGA (640x480) : 30 fps QVGA (320x240): 60 fps
Resolution	VGA (640x480)	VGA (640x480)	SXGA (1280*1024)
Operation Environment	Indoor	Indoor	Indoor
Dimensions	28.5 x 6 x 7.5	28.5 x 6 x 7.5	18 x 3.5 x 5 cm

5.1.1.1 Hardware Layer

Examples of consumer grade depth sensors that are available to the public at the time of writing are the Asus Xtion Pro Live and the Microsoft Kinect for Xbox and Windows. Brief specifications of each device are shown in Table 5.1. All devices offer both RGB and depth sensors, however, the Asus Xtion Pro Live provides higher resolution for the RGB image. The Asus Xtion Pro Live also provides a lower resolution depth image at a higher frame rate of 60 fps. All sensors have similar operating ranges except Kinect for Windows, which can determine distance as close as 40 cm from the device.

5.1.1.2 Interface Layer

This layer encapsulates different software drivers and APIs provided by the hardware developers or community efforts, and provide a standard interface to the higher layers. As of writing the two main APIs available are the Kinect SDK (Webb & Ashley, 2012), which interfaces with the Xbox and Windows Kinect, and the OpenNI library (Falahati, 2013) which interfaces with the Asus Xtion Pro Live. At the base level both libraries provide access to RGB and depth images generated by the sensors, but also offer higher functionalities as well. The Kinect SDK offers full-body skeletons tracking, face tracking, and a speech recognition API, while OpenNI offers full-body skeleton tracking. Note that these higher level functionalities were not utilized in this research.

5.1.1.3 Retrievers Layer

This layer encapsulates existing libraries that provide wrapper classes for instantiations of sensor nodes, such as OpenCV (Bradski & Kaehler, 2008),

Point Cloud Library (PCL) (Rusu & Cousins, 2011). A custom retriever is also available for flexibility, allowing for more control over retrieval management such as identifying the grabber's id, starting and stopping individual retrievers etc.

5.1.1.4 Manager Layer

When multiple sensors are used, multiple retrievers must be instantiated and managed by a central process, the retrieval manager. The manager provides an abstraction layer that provides a list of all available devices such that the user can easily instantiate the correct retriever based on the device id.

5.1.1.5 Outputs Layer

The output layer provides a standard data interface to the higher layers, and is extensible such that new data types can be added as needed. Currently, three types of data can be provided; the color image and depth image in OpenCV's mat or IplImage format and color point clouds in PCL's PointCloud<PointXYZRGB> format.

5.1.2 Segmentation and Tracking

The segmentation and tracking layer provides functionalities that can be categorized into three sub-layers; (1) Segmentation, (2) Identification and (3) Tracking. These layers operate on the input images and point clouds using color, depth and spatial properties such as location, shape and size. The illustration of this layer is as shown in Figure 5.3.

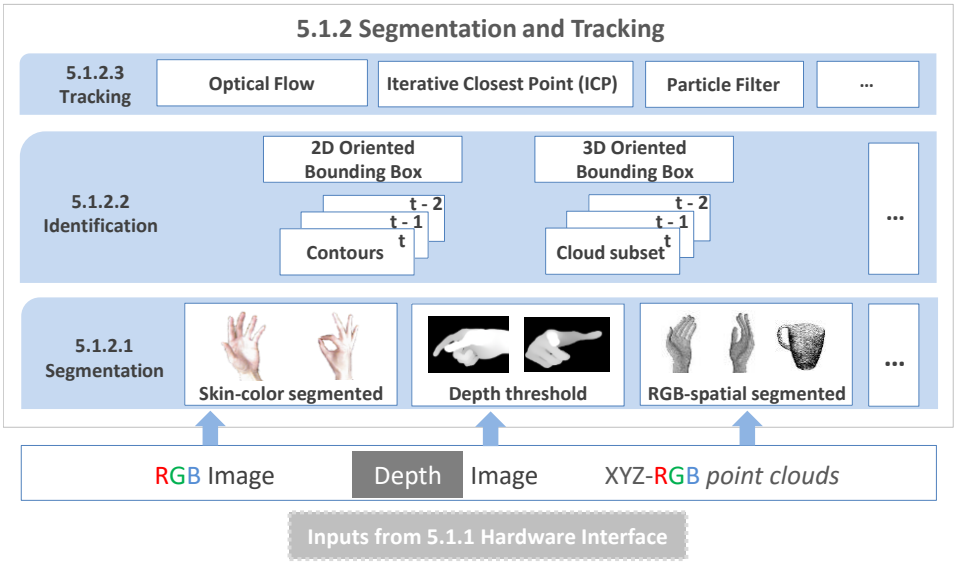


Figure 5.3: The segmentation and tracking layer

5.1.2.1 Segmentation

With the input data provided from the hardware interface, the segmentation module provides functions for filtering the RGB and depth images or XYZRGB point clouds according to developer defined criteria. In the current implementation, the available filters include an RGB filter, skin-color filter using a Gaussian Mixture Model, depth threshold using a given depth value and location filter. Each filter uses the most appropriate input format, for example the skin-color filter accepts color images or point cloud data with an RGB

channel, and the depth threshold filters takes floating point depth images or point clouds. For 2D images, all filters return a 2D mask image, where a zero value represents pixels of interest, allowing for easy processing of the original image. In the case of the point clouds, all filters return a new point cloud set that is filtered to only include points of interest.

5.1.2.2 Identification

After segmenting, objects found in a 2D image are represented by contours, while objects found in a 3D point cloud are represented as a subset of points. For each object detected in the current frame, a fitting ellipse based on the principle axes is calculated. If the object is a contour then the principle axes and the resulting oriented bounding box (OBB) computed are 2D, and if the object is a point cloud then the principle axes and the OBB are 3D. The OBB information is used for reorientation of the object of interest for further processing. This is useful for algorithms such as the hand region classification algorithm in Section 5.1.3 which is not rotation invariant.

The OBB provides an approximate location of the object's center in the current frame, and by comparing the data from past frames, the OBB can be utilized for tracking object movement between frames. Each object's current state is compared to the list of existing objects from the previous frame, and the objects with the closest matching state are assumed to be the same object across different frames. If the number of objects in the current frame changes from the previous frame then an object should be created or removed accordingly. Averaging across frames can also reduce any noise in the segmented data or OBB.

5.1.2.3 Tracking

The tracking module encapsulates several algorithms for estimating the movement of the object of interest. For 2D image data, optical flow can be used to determine a translational vector for each pixel, and this can be extended to a three-dimensional flow image if the corresponding depth image is included.

For point cloud data, iterative closest point (ICP) or particle filter algorithms can be utilized. ICP attempts to match sets of point clouds, minimizing the spatial difference between them by iteratively estimating the transformation between the corresponding points. Particle filters work using, a probability distribution of the object's state, such as location, size etc., which are represented by particles. Each particle represents one probable state where the weight determines the probability that it is the correct state.

These algorithms provide a robust approximation of the object's state that updates over time, yielding a more accurate model the user's actions.

5.1.3 Hand Regions Classification

Our hand region classification method is comprised of six steps; (1) Synthetic hand poses creation, (2) Decision tree training using GPU, (3) Hand segmentation, (4) Decision forest classifier using GPU, (5) Post-processing, and (6) Estimating joint position. Steps 1 and 2 are pre-processing steps and used to generate the necessary decision tree training data offline, while Steps 3 to 6 are executed online to classify the input image in real-time. The data flow diagram of this method is shown in Figure 5.4.

5.1.3.1 Synthetic Hand Poses Creation

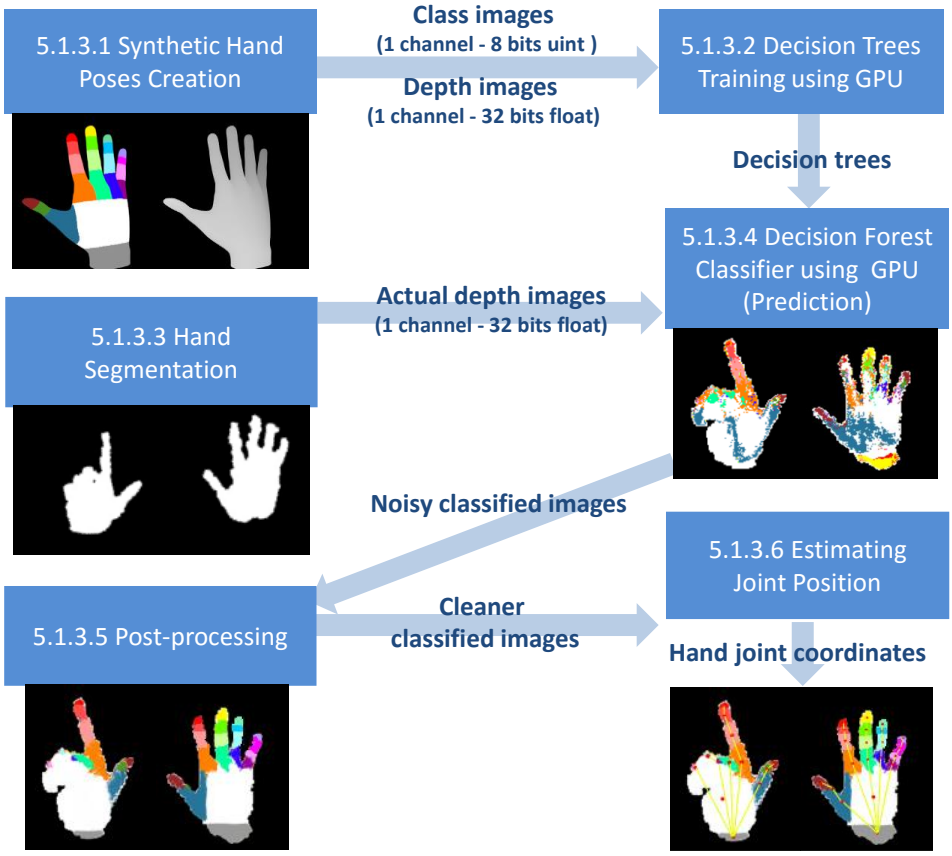


Figure 5.4: Data flow in hand classification

The human hand is made up of 27 bones: 14 phalanges which constitute each finger, 5 metacarpals connecting each finger to the wrist and 8 carpals located in the wrist itself (see Figure 5.5a).

We used Autodesk 3ds Max, a 3D modeling, animation, and rendering software application, to create a synthetic hand model and generate color and depth images of hand poses. The approximated model consists of 17 bones, 16 joints and 21 unique regions (see Figure 5.5b and Figure 5.6a), and hand mesh deforms according to its assigned bone position. This constitutes a hand model

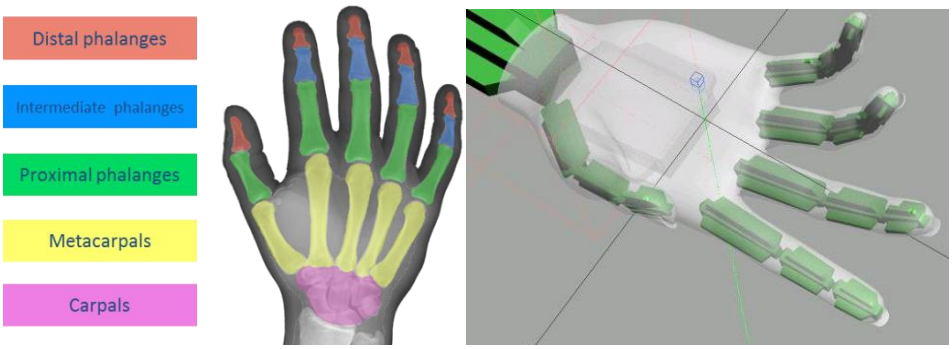


Figure 5.5: (a) Hand's anatomy (b) 3D hand model in 3DS Max

with 27 *dof* was first introduced in (Jintae & Kunii, 1993). This model can approximate most hand postures and is sufficient for hand classification.

The hand is divided into 21 unique regions such that the center of each region falls on each joint's location, the center of the palm or the distal phalanges. One joint on the thumb located between the metacarpus and carpus, is omitted as the range of motion offered by this joint is insignificant for hand classification. After classification, if all regions are visible, a maximum of 21 fixed points can be calculated and be connected hierarchically into a hand's skeleton as shown in Figure 5.6b.

Approximately 110,000 color and depth images were generated by manually creating 400 unique hand postures and then capturing these postures from 275 unique camera angles. The resulting color images were generated from the RGB channel and the depth images were generated from the z-depth channel, both at 160 by 120 pixels.

5.1.3.2 Decision Trees Training using GPU

A. Depth features

Due to the simplicity and computational efficiency, the same depth comparison features have been adopted from (Jamie Shotton et al., 2011) and (C. Keskin, F.

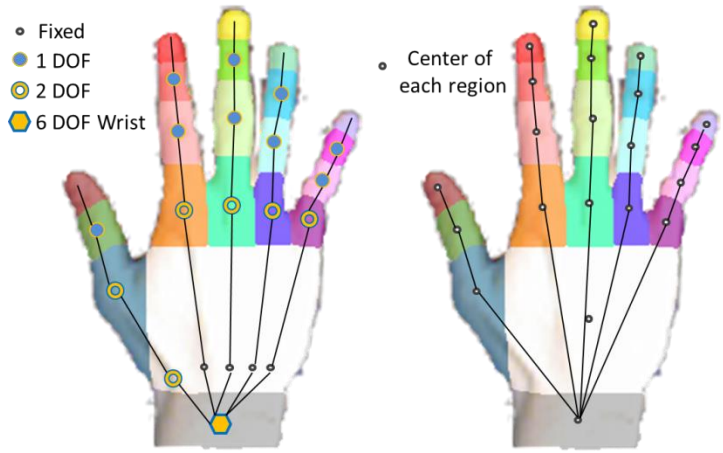


Figure 5.6: (a) 27 DOF hand model (b) Hand skeletal estimate

Kirac, Y. E. Kara, & L. Akarun, 2011). The chosen features are translation invariant but not invariant to rotation and scale. The assumption used is that the segmentation step will give us an input depth image where non-zero pixels represent the hand and zero pixels are the background. The features, $F_{u,v}(I,x)$, can be calculated by taking the difference between the depth at the offsets u and v from pixel x as shown below:

$$f_{u,v}(I,x) = d_I\left(x + \frac{u}{d_I(x)}\right) - d_I\left(x + \frac{v}{d_I(x)}\right) \quad (5.1)$$

Given that $d_I(x)$ is the depth value at pixel x in the input depth image, I. The offsets u and v are normalized by division by depth at x. If the offset pixel falls on the background, the feature value is set to a large constant value of 10,000.

B. Training random forest

Random forests (Breiman, 2001) are made up of multiple decision trees trained with unique randomly selected images from the training set created as described in Section 5.1.3.1. The tree consists of split and leaf nodes, where a split node has attributes of a feature $f_{u,v}$ and a threshold τ and the leaf node contains the

predicted class and its probability. The algorithm for training each tree is described as follows:

1. Randomly sample pixels with non-zero value from each image where the maximum number of sampled pixels is set to be $\frac{(image_{width} * image_{height})}{10}$. Load all images data into GPU memory.
2. Create 4K tuples of $\{u, v, \tau\}$ combination where the offsets u and v are chosen between 0 to 60 pixels and τ from -200 to 200 mm, according to (C. Keskin et al., 2011).
3. Split the dataset $Q = \{(I, x)\}$ into left and right subsets for each $\{u, v, \tau\}$ combination.

$$Q_l(u, v, \tau) = \{(I, x) | f_{u,v}(I, x) < \tau\} \quad (5.2)$$

$$Q_r(u, v, \tau) = \{(I, x) | f_{u,v}(I, x) \geq \tau\} \quad (5.3)$$

   On the GPU, for all sampled image pixels from step 1, compute the depth feature $f_{u,v}$ and compare to the threshold value τ concurrently.
4. Find the $\{u, v, \tau\}$ combination that gives the largest information gain.

$$G(u, v, \tau) = H(Q) - \sum_{s \in \{l,r\}} \frac{|Q_s(u,v,\tau)|}{Q} H(Q_s(u, v, \tau)) \quad (5.4)$$

   Where $H(Q)$ is the Shannon entropy computed on the normalized histogram of classes in dataset Q. On the GPU, for all the $\{u, v, \tau\}$ combinations, compute the information gain concurrently and find the combination that produces a maximum score $\{u, v, \tau\}_{max}$. This combination is taken as the split condition at the current split node in the tree.
5. If the maximum tree depth hasn't been exceeded and this node isn't a leaf node, then repeat step 3 for the left subset $Q_l(\{u, v, \tau\}_{max})$ and right subset $Q_r(\{u, v, \tau\}_{max})$.

5.1.3.3 Hand Segmentation

The depth image that was segmented as described in Section 5.1.2 is taken as an input for classification. Since this method is neither rotation nor scale invariant, we pre-process the image to ensure rotation and scale effects are removed. First, each hand object is identified and tracked. Second, scaling is applied to create an image with a constant hand size. Third, the image is oriented such that the hand is always oriented the same direction. Finally, as the training data only contains right-handed poses, any left-handed images are flipped prior to the prediction.

5.1.3.4 Decision Forest Classifier using GPU

An ensemble of decision trees generated as described in Section 5.1.3.2, makes up the random forest. Given the input depth image generated in Section 5.1.3.3, each pixel is classified independently. Given a non-zero pixel, the classification begins at the root of the tree and propagates down each split node until reaching a leaf node. At each split node, Equation 5.1 is calculated with the attributes, $\{u, v, \tau\}$. The offsets u and v are substituted and the resulting $f_{u,v}(I, x)$ is compared to the threshold, τ. If the resulting feature is below the threshold then the algorithm traverses to the left child, otherwise it traverses to the right. When a leaf node is reached, the class and its distribution $P_n(c \mid I, x)$ is stored. The final classification result is determined by taking the average value over all trees in the random forest, as shown in Equation 5.5.

$$P(c \mid I, x) = \frac{1}{N}\sum_{n=1}^{N} P_n(c \mid I, x) \qquad 5.5$$

Since the classification of each image pixel is independent from every other pixel, this step can be implemented on a GPU for parallel classification. Furthermore, as independent classifications must be performed on multiple decision trees, each trees classification can be parallelized for the same pixel as

well. This allows for parallelization of classification for all image pixels among all trees at the same time, providing significant performance improvements.

5.1.3.5 Post-processing

One downside of this approach is that the resulting classified image is usually noisy. A post-processing algorithm is used to remove this noise. The steps taken are as follows:

Step 1: Estimate the location of the wrist by finding the narrowest cross-section in the possible area where the wrist could be located within the image. First find the principle axes for an area classified as an arm region (Class 1), then find the narrowest perpendicular strip of pixels along the longer axis. Take the mid-point on this strip as the wrist's center. Note that this rule cannot be applied when the user's arm is parallel to the cameras z-axis, and this condition is checked before calculation.

Step 2: A large area of pixels usually indicates the user's palm (Class 2). During pre-processing the hand image is reoriented and scaled, and as such class 2 pixels usually cover a significant portion of the central area of the image. When a large area of class 2 pixels is identified, any small collections of non-class 2 pixels in this area can be reclassified as class 2. The result is a single blob of class 2 image identified as the palm.

Step 3: From observation, classification of the thumb is typically robust and accurate. Therefore we use the thumb as a reference for correcting the other hand regions. When the thumb is found then we find the closest finger that is visible in the image, which should be the index finger, and from this we can deduce the direction the hand is facing. The other hand regions can be enforced from this deduction.

Step 4: Finally, for each pixel we examine, the classes of its nearest neighbors based on position in the depth image, and calculate the majority class in this neighborhood. Each class is weighted given its importance to hand classification, for example the classes belong to the distal phalanges region are weighted higher than the intermediate phalanges and proximal phalanges. When considering the nearest neighbor, the depth data must be normalized by taking the ratio of the depth value and the range of depth value in this image (See Figure 5.7).

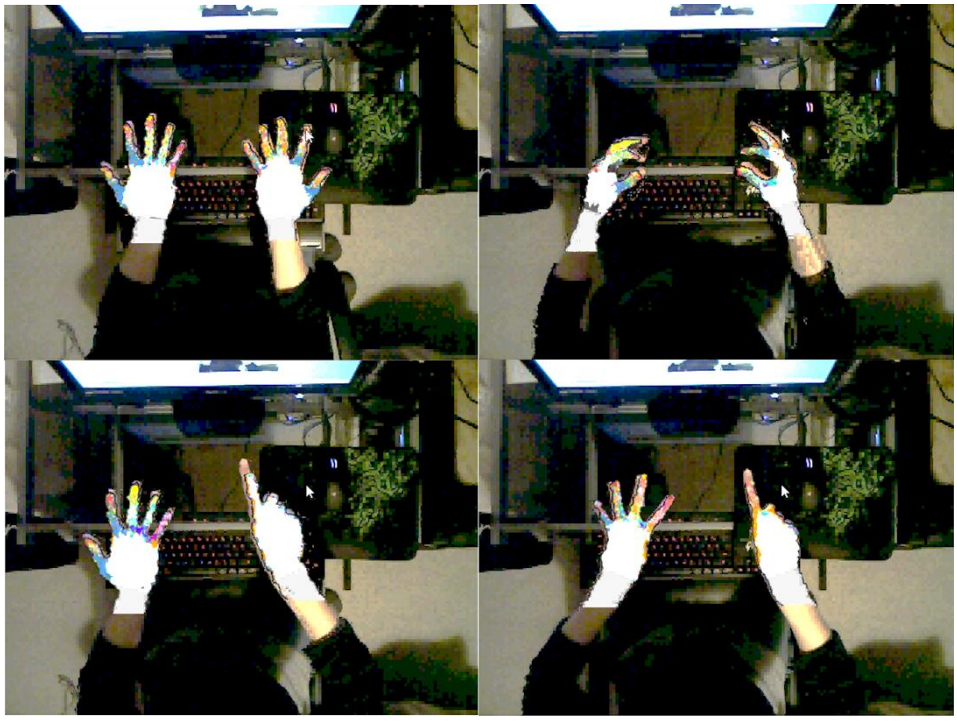


Figure 5.7: Sample results of hand tracking and classification

5.1.3.6 Estimating Joint Position

After post processing, the hand is effectively divided into regions, where each region only has a single class. With this, we calculate the centroid of each

visible region and set this as the joint position. From the visible regions and the overall direction of the hand and fingers, the centroid of each unknown region can be estimated based on the hierarchical structure of the hand.

5.1.4 Modeling

The modeling layer takes output from either the segmentation or the classification layers. Modeling has been categorized into three categories; (1) Hand modeling, (2) Object modeling, and (3) Simplified physics representation. These representations can be used for collision detection as well as simulating the dynamics. The detail of each category is described as follows.

5.1.4.1 Hand Modeling

This category is dedicated to modeling the hand, and can be further sub-divided into skeleton-based and model-based.

5.1.4.1.1 SKELETON-BASED

After classification of the hand, we have the 3D positions of the 21 hierarchical joint positions, as shown in Figure 5.6b. A simple hand model composed of geometric primitives such as spheres, cylinders and boxes, can be used to represent the hand. The primitive shapes are transformed based on the relative joint positions, and this simple approximation offers a computationally efficient approximation of the hand.

5.1.4.1.2 MODEL-BASED

In model-based hand model, the hand is represented by an accurate skinned mesh with bones and pre-defined constraints for the joint parameters, similar to Figure 5.6a. All joint propositions must be tested to ensure the constraints aren’t

violated. Compared to skeleton-based modeling, this method provides more accurate model at the cost of computational time.

5.1.4.2 Object Modeling

This module allows for modeling tracked pre-defined known rigid objects. The model is transformed by applying the transformation that is found in the segmentation layer.

5.1.4.3 Simplified Physics Representation

This module provides a simplified physics representation of unknown objects that aren't hands. Although it provides a lower accuracy geometric representation, it is a good approach for approximating real-world objects (T. Piumsomboon et al., 2012). These objects can either be represented using Spherical Proxies, or by Mesh Reconstruction.

5.1.4.3.1 Spherical Proxies

This technique represents the visible surface of a real object using spheres, as discussed in Section 3.1.2.3. This is a popular technique as it provides approximate modeling in a computationally efficient way (Hilliges et al., 2012; Benko et al., 2012).

5.1.4.3.2 Mesh Reconstruction

As described in Section 3.1.2.2, mesh reconstruction provides a simple estimate of the visible surface where sampled points on the surface are triangulated into a mesh.

5.1.5 Posture and Gesture Recognition

After classification as described in Section 5.1.3, we know the 3D locations for 21 unique joint positions, which can be used to train a hand postures recognizer. We originally planned to use the training sets with two sets with different number of joints which are 10 and 21 joints. The training was to be done for 2D and 3D space, which would have resulted in a dataset with 20 and 30 dimensions for 10 joints, and 42 and 63 dimensions for all 21 joints using a support vector machine (SVM) as a classifier. Unfortunately, the large size of the dataset and significant error made this an unsuitable choice, and a more simple recognizer, the $1 recognizer (Wobbrock et al., 2007) was used instead. This recognizer is simple, computationally cheap, and usable, provides position, rotation, and scale invariance and offers high accuracy with small number of loaded templates. The performance is comparable to the use of dynamic programming and statistical classification.

5.2 Discussion

In this section, the contributions and state of the interface, performance, and limitations are discussed in Sections 5.2.1, 5.2.2, and 5.2.3, respectively.

5.2.1 Contributions and State of Interface

Our main contributions in this interface includes (1) implementation of a hardware interface using OpenCV and Point Cloud Library (PCL), (2) implementation of hand segmentation and tracking, (3) hand region classification where training is performed on a GPU, (4) modeling on a physics engine, and (5) integration of a $1 recognizer to gesture recognition.

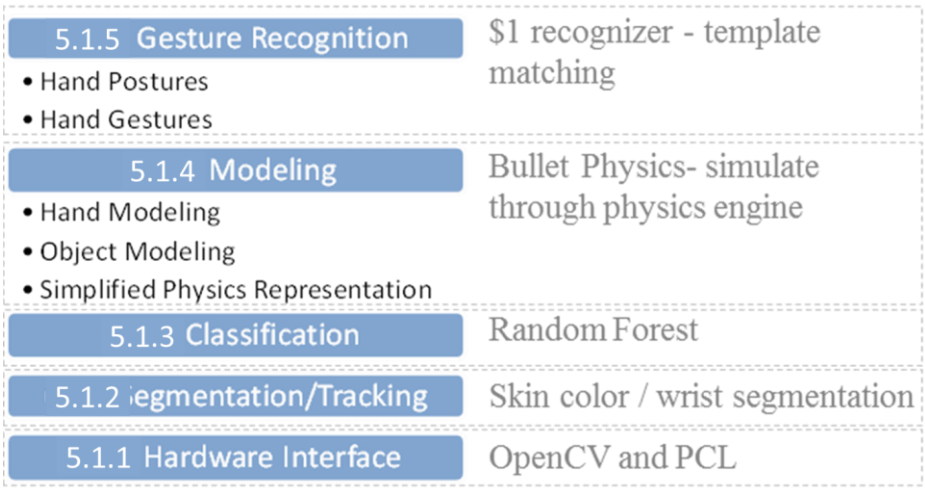


Figure 5.8: A brief description of the gesture interface implementation

Figure 5.8 illustrates the current state of the gesture interface, where each component is modular and can be replaced with another solution as needed. In the hardware interface layer, OpenCV, an image processing and computer vision library that focuses on 2D image input, and PCL, a 3D computer vision library that is based on point clouds, are used to interface with the sensing device. The processed depth and color images are passed to the hand tracking and segmentation layer where skin color segmentation is used to distinguish hands from the background and the oriented bounding box for each hand is found. With the segmented depth image of the hand, random forest classification is applied to identify different regions of the hand, and subsequently, fingertips and joints are found. The hand modeling layer can use the information obtained about hand regions to simulate hand behavior in a physics simulation engine. The same data can be used for hand gesture recognition, with the present implementation using the $1 recognizer due its simplicity and fast computation time.

5.2.2 System Performance

The computer used for the development was an Intel Core i7 3.2 GHz with one Nvidia GTX 680 and two GTX 580 graphics cards. The time taken to train a decision tree can vary considerably depending on parameters such as the number of images, tree depth etc. A tree depth of ten layers, training on a single GTX 580 takes approximately an hour. The segmentation, tracking, classification, modeling, and recognition process are performed on CPU, and can run at 20 FPS using a random forest of 20 trees. The computational cost for modeling and recognition are negligible compared to the cost for classification.

5.2.3 Limitations

Hand tracking and classification were the most challenging components to develop and the prediction accuracy is still limited. There are situations that cause failures in tracking and error in classification, such as when random forest classification yields results with a high level of noise.

Tracking and classification within this interface is currently capable of simple recognition, for example static hand poses that do not require differentiating from frame to frame with a high level of precision. Nonetheless, our interface is extensible, which allows for improvements to provide continuous and precise hand pose estimation and hand tracking and classification.

5.3 Lessons Learnt and Recommendations

There were a number of challenges that had to be overcome when developing our hand tracking and pose estimation components. These challenges included (1) determination of the initial position of the hands, (2) training with the synthetic hand training dataset, (3) discovering appropriate parameter values for

training, (4) high sensitivity of the classification to rotation and scaling of the input image, and (5) the most challenging components to develop, are in tracking and classification. The details of each challenge are as follows:

5.3.1 Initialization of Hand's Position

For our interface, we only track hands without any other information about the body, however tracking the other parts of the body that connect to the hand may improve the initial prediction For example, in our setup, the camera was positioned from the top pointing down onto the user's head. Given that the head is in the camera view, it should be possible to determine the user's head, shoulder, and arm positions, and from this estimate the initial position of the hands.

5.3.2 Limit the Scope of the Training Set

From our experience generating a synthetic hand dataset for training, we feel it may be better to focus on a subset of all possible hand poses and orientations. It may also be beneficial to vary the hand model size for robustness.

5.3.3 Trial and Error in Finding Optimal Training Parameters

Trial and error was used to determine the best parameters, such as the tree depth and the neighboring pixel distance, when training each tree. Training is a time consuming process, and it would be beneficial to perform this task on multiple servers to evaluate several parameter sets at once.

5.3.4 Sensitivity to Rotation and Scaling of Input Image

The proposed classification technique is sensitivity to rotation and scaling, and the accuracy depends on how accurately the input image is oriented and scaled

prior to classification. We suggest using the wrist position and the hand direction to calculate the most accurate result. In addition, adding slight changes to hand pose angle into the training set may give better accuracy.

5.3.5 Hand Tracking and Classification are challenging

To conclude, hand tracking and classification were the most challenging components of the gesture interface to develop and the prediction results are still limited. However, promising research and development is ongoing and depth sensing technology is still the best hardware solution for gesture interfaces.

Due to these limitations, for our final AR framework we opted to use a commercial hand tracking and classification solution, Nimble SDK, to improve the user experience. The final framework is presented in Chapter 6: Multimodal Augmented Reality Framework and Interaction Techniques.

5.4 Conclusion

This chapter presents our gesture interface, which is comprised of five major components including; (1) a hardware interface that grabs inputs from the source device, (2) a segmentation and tracking component that detects the wrist position and orientation of the hand in the scene, (3) classification using random forests for identifying hand regions and subsequently the hand joints, (4) physical modeling to support physics enabled simulation, and (5) gesture recognition that employs a simple but efficient $1 recognizer. We discussed the limitations of this system and lessons learnt during development, and offer recommendations for future improvements.

We found the hand tracking and classification component the most challenging to develop, with the random forest algorithm yielding high levels of noise in the resulting classification images. As in this thesis we are primarily

interested in the effectiveness and preference of interaction techniques as opposed to the technical development of underlying components, in our AR framework we replaced our tracking and classification component with a commercial solution. This meant that we could focus on user experience without the user being hindered by limitations in the technology. As our framework is extensible, future improvements and extensions to the hand tracking and classification component can be implemented to match or even exceed the performance of these commercial offerings.

In the next chapter we describe the integration of the software components that we have developed in previous chapters into our AR framework, G-SIAR. We also present two interaction techniques implemented into this framework based on this gesture interface, a direct natural hand interaction technique, Grasp-Shell (G-Shell), and an indirect multimodal gesture-speech interaction technique, Gesture-Speech (G-Speech).

Chapter 6
Multimodal Augmented Reality Framework and Interaction Techniques

The main goal of this research is to explore novel natural hand interaction as a primary input for AR and validate its usability compared to gesture-speech input. In the previous chapters we have explored and implemented a number of required components for a multimodal augmented reality framework, starting with physically simulated and environmentally aware systems as described in Chapter 3, followed by natural hand interaction in AR and gesture input in Chapters 4 and 5 respectively. In this chapter we bring these features together in a new framework for AR with gesture-speech integration and natural interaction as a native input.

Within this framework, we present two natural interaction techniques, a direct natural hand interaction and an indirect gesture and speech interaction. We define direct and indirect interaction as either requiring the user to make contact with the manipulated objects or allowing the user to perform a task from a distance respectively. By supporting two complementary interaction techniques in our framework, their usability can be tested and compared. However, a more useful purpose is to demonstrate the strengths and weaknesses of different methods of interaction, where we believe that neither technique is universally better as different levels of directness of interaction will be better suited to different tasks.

In this chapter, Section 6.1 introduces the framework, and Section 6.2 describes the design and implementation of the framework. The two interaction techniques implemented in the framework are discussed in Section 6.3. Section 6.4 discusses the performance of the system and Section 6.5 summarizes lessons

learnt from the development of this framework and interaction techniques. The chapter is concluded in Section 6.6.

6.1 Introduction

Recent research in interaction has suggested that high levels of interactivity and precision in AR can be achieved through natural interaction (Benko et al., 2012; Hilliges et al., 2012). These papers demonstrate that hand based interaction can benefit application in areas such as interactive games, 3D modeling, rapid prototyping, and remote collaboration.

In previous chapters we explored some of these interaction techniques, and we now present our custom interaction framework, G-SIAR (Gesture-Speech Interface for Augmented Reality, pronounced "g-seer"), that uses gesture and speech as the primary inputs, provides visuals cues such as shadows and hand occlusion, and supports a physics-enabled environment in AR. The AR experience is experienced using a wide field-of-view (*fov*) video see-through head-mounted display made by mounting wide angle stereo cameras on the Oculus Rift display.

By utilizing cutting edge hand tracking and speech recognition technology, higher precision hand gesture and speech input is possible. In this research, our focus is on designing novel natural hand interaction techniques based on our findings discussed in previous chapters. This led to the development of two interaction techniques: a natural hand interaction technique named Grasp-Shell (G-Shell) and a multimodal technique named Gesture-Speech (G-Speech). A number of hand gestures from the user-defined gestures set found in Chapter 4, were implemented, with support for multiple variants of hand poses.

G-Shell is a novel natural hand interaction technique that is differs from earlier research in a number of ways. It uses 6 *dof* hand tracking technology to

create a natural, intuitive and interactive experience. It also introduces the concept of a "shell" in physical simulation for collision detection. Using this approach, precise direct manipulation of virtual objects is made possible.

G-Speech is a multimodal gesture-speech interaction that offers deictic gesture and metaphoric gestures. We implemented G-Speech following the design recommendation from the WOz study (Minkyung Lee & Billinghurst, 2008) and the guessability study in Chapter 4.

This chapter has two main contributions: (1) The design and implementation of an interactive system that uses natural hand interaction, and gesture and speech as the primary inputs in AR and (2) The design and implementation of the G-Shell and G-Speech interaction techniques.

6.2 Design and Implementation of G-SIAR

In this section, an overview of a custom built interactive framework for AR called G-SIAR is given. The hardware and software implementations are discussed in Sections 7.3.1 and 7.3.2, respectively. The general design goals of G-SIAR that was set out to achieve were:

i. Use gesture and speech as the primary inputs for AR interaction.
ii. Provide an interactive and precise environment that can support a wide range of applications.
iii. Offer experiences across the Mixed Reality continuum from AR to VR.

6.2.1 G-SIAR Hardware

The hardware design goals were as follows:

i. Support natural hand tracking.

ii. Provide highly immersive and large viewing coverage of the interaction space.

iii. Support transitions across the reality-virtuality continuum from AR to VR.

The hardware used in our current system includes (See Figure 6.1):

A. AR-Rift (Oculus Rift HMD mounted with wide-angle stereo cameras)
B. PrimeSense Carmine 1.09 (depth sensor)
C. Creative Senz3D camera (depth sensor)
D. Image-based marker (A1 paper size)
E. Alienware 17 laptop (Intel Core i7-4800MQ @ 2.70Ghz with Nvidia GeForce GTX 780M)

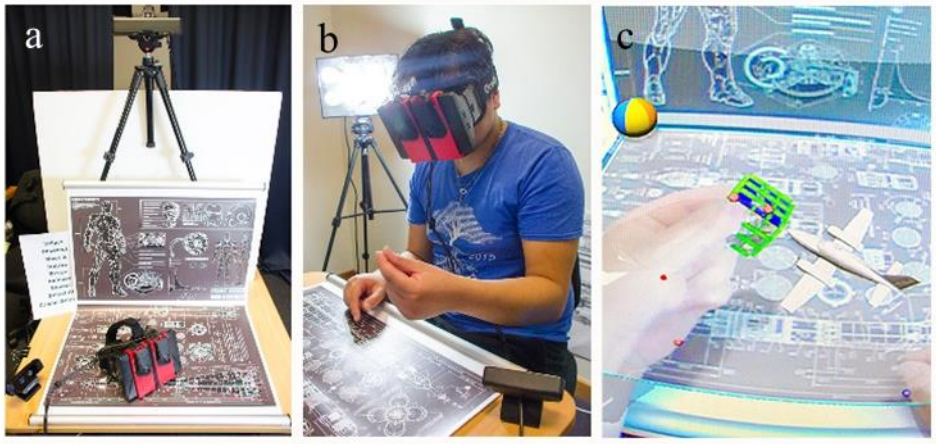


Figure 6.1: (a) System setup, (b) A user is grasping onto a tiny virtual rubik's cube, (c) The user's view

6.2.1.1 AR-Rift, Customized Video See-through HMD

The Oculus Rift was chosen as a display device due to its large Field of View (*fov)*. By attaching wide angle stereo cameras to the Rift, a highly immersive user experience can be delivered across the whole Mixed Reality (MR) spectrum (Milgram & Kishino, 1994). The AR-Rift implementation was based on the design of Steptoe (Steptoe, 2015), who created a wide-angle stereo camera setup that was compatible with the display properties of the Rift. Two Logitech C270 cameras with an 800 x 600 pixel resolution were used as the camera for the AR video feed. The Logitech C270 lenses were replaced with Genius WideCam F100 wide-angle lenses, resulting in a horizontal and vertical *fov* of approximately 116° and 94° respectively. By attaching the cameras horizontally, video was captured at 600 x 800 resolution with a 3:4 aspect ratio. Each video image was padded horizontally with a 20 pixel borders and shifted depending on the user Inter Pupil Distance, resulting in a display image of 640 x 800 for each eye matching the display resolution of the Rift. The user's view of the AR-Rift is illustrated in Figure 6.3.

Figure 6.2: Two versions of AR-Rift

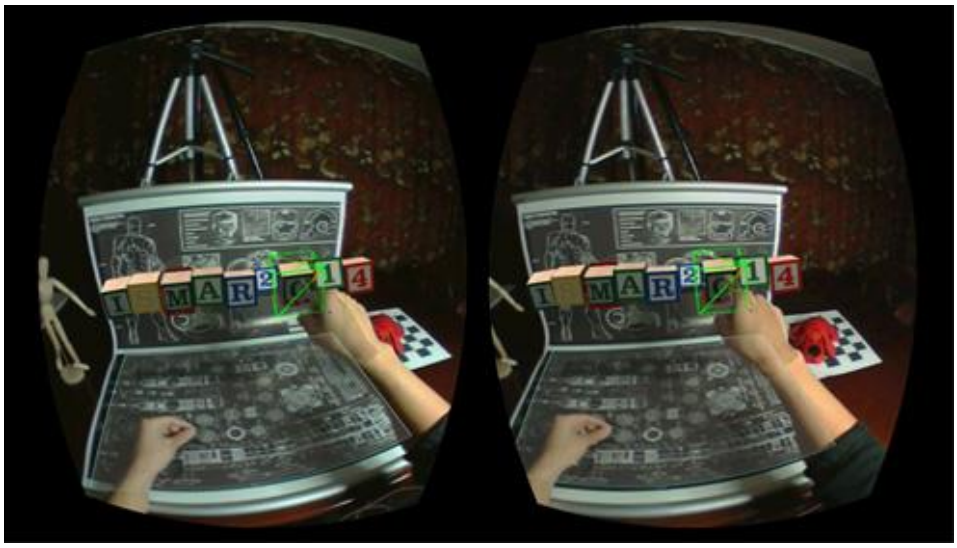


Figure 6.3: The view in the AR-Rift

Camera mounts and protective covers were custom designed and 3D printed. Two versions of the AR-Rift were developed, one matching Steptoe's original configuration, and another with a depth sensor mounted on top as shown in Figure 6.2. Further discussion of the hardware can be found in Section 7.2.2.5.

6.2.1.2 Interaction Space

The 3D structure of the environment was captured using a PrimeSense Carmine 1.09 depth sensor. This was positioned 700 mm above the interaction surface on a tripod, and pointed down towards the space. The range and *fov* of the depth sensor defined the size of our interaction space: 600 x 450 x 450 mm (width x depth x height). An image-based marker was created to allow for positional tracking of the AR-Rift (see Figure 6.1), and to provide a visual boundary of the interactive space for the user. The high quality microphone array in the Creative Senz3D camera was used to capture speech commands.

6.2.2 G-SIAR Software

The design goals of the G-SIAR software were as follows:

i. High *dof* hand pose estimation for natural hand tracking.
ii. Support for physics-enabled simulation with high accuracy.
iii. Support for real-time and seamless object creation and interaction.
iv. Support for realistic rendering with shadows, hand occlusion, and distortion for displaying within the Rift.

6.2.1.1 Architecture Overview

Figure 6.4 illustrates a simplified architecture of G-SIAR. Initially our plan was to use our custom Gesture Interface designed in Chapter 5 for hand tracking, however we felt the lower than ideal accuracy of the interface would be distracting to users, so instead we chose to use a commercial product, the 3Gear Nimble SDK.

Inputs into the G-SIAR framework include hand pose data from the Nimble SDK, audio and depth images captured by the Creative Senz3D, and dual video streams from the AR-Rift stereo cameras. The output includes visual feedback through the AR-Rift and audio feedback through speakers or headphones.

To ensure the best response time, G-SIAR is multi-threaded with every module running in its own thread. The key software components include 3Gear's Nimble SDK (hand tracking), OpenSceneGraph (graphics), GLSL (shader), Bullet (physics), OpenCV (image processing), OsgBullet (graphics and physics interface), OPIRA (marker tracking), Intel Perceptual SDK (camera capture and speech recognizer), FMOD (audio output), Oculus SDK (display), and Boost (threading and data structure).

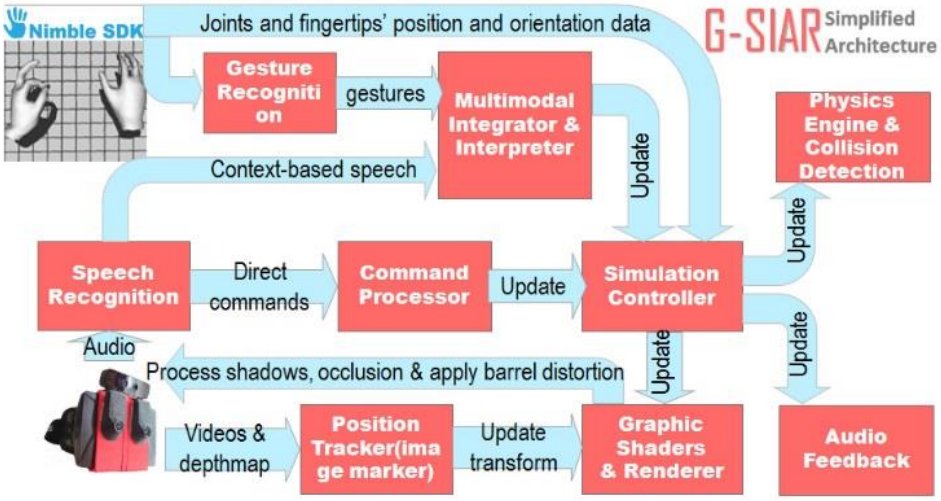


Figure 6.4: G-SIAR architecture

In the remaining sections of this section, each component is described in more detail.

6.2.1.2 Gesture Recognition

The key to higher precision natural hand gesture interaction is high accuracy and high *dof* hand pose estimation. The framework we designed in Chapter 5 fell short of these requirements, so a commercial framework was used for our study. The 3Gear Nimble SDK, based on the research of 6D hands (R. Wang et al., 2011), supports six *dof* bimanual manipulation. It provides very accurate six *dof* natural hand tracking; tracking the wrist, 15 joints, and 5 fingertips for both hands with millimeter-level precision for every finger at 30 fps.

The Nimble SDK provides gesture detection for pinch and pointing gestures, however there were limitations to this. For example, the pinch action requires that the hand faces the camera so that the hole made by the thumb and finger is visible, limiting the angle at which a pinch can be detected. To overcome this, the G-SIAR system defines its own a pinch action as when the

Euclidean distance between the finger and thumb is smaller than a threshold defined on a per user basis. Another limitation of Nimble was a noticeable delay in pointing detection, which was resolved by changing the classification to use a comparison of the Euclidian distance between the fingers and a stable reference point such as the wrist, which resulted in almost instantaneous detection.

6.2.1.3 Direct Inputs and Multimodal Inputs

Gesture and speech can be used for both direct and multimodal input. Speech can provide direct input in the form of commands that do not require context, such as “enable gravity” to turn on gravity in the simulation. Gesture input can provide hand position and orientation for direct physical interaction, for example physically pushing objects with the fingertips. For multimodal input, G-SIAR features a multimodal integrator, which is responsible for determining the action required based on combined speech and gestures. The integrator acts when a verbal command is given that requires gestural context. For example when the user points at an object and says “change the color to red”, the integrator informs the interface controller of the action, the selected object and the color information to change the color of the selected object to red.

6.2.1.4 Contacts Points and Dynamic Simulation

In AR, physics-based simulation can enhance user experience in terms of realism and believability, and can also improve usability. In the real world, when a user grasps an object, interaction between the fingertips and the object is initiated at the contact point(s). In G-SIAR, the user’s fingertips are tracked using a depth sensor and the physics engine’s collision detection algorithm is

used to monitor the occurrence of contact in 3D space. Once this is known, it is possible to interpret the user's intention and complete the user interaction.

6.2.1.5 Shaders for Shadows, Occlusion, and Distortion

Graphical shaders were used to provide realistic hand occlusion, shadows, and the required distortion for the Oculus Rift display. The rendering framerate was approximately 60fps. To allow this high frame rate, several techniques were explored, as described in the following sections.

6.2.1.5.1 Skin segmentation and per-pixel occlusion

Depth data from the user's perspective was provided by the Creative Senz3D mounted on top of the AR-Rift. This data is processed using a skin color segmented image to create a depth map only containing values that belong to the hands. This processed depthmap is passed to a shader that tests the depth at each pixel, and if the hand pixel is closer than the pixel to render the background texture of the hand is shown instead (see Figure 6.5a).

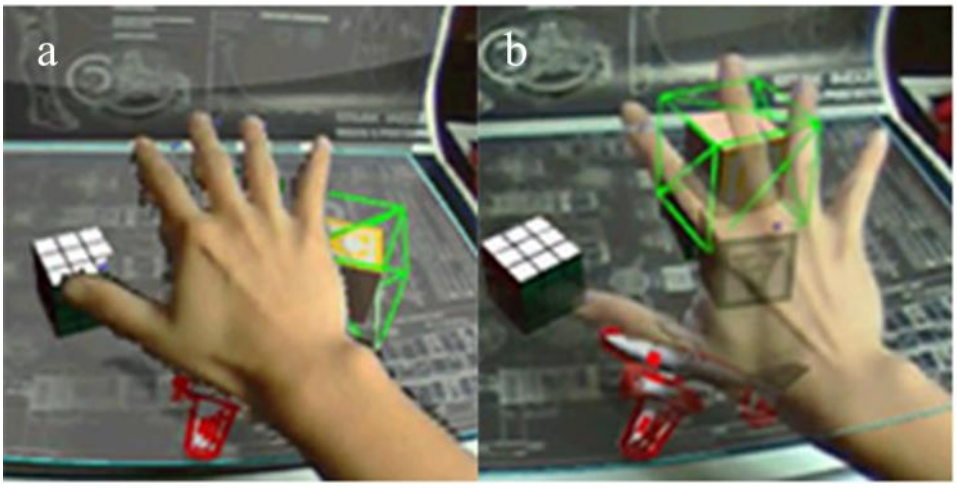


Figure 6.5: (a) occlusion of method A and B, and (b) Hand occlusion of method C

The advantage of this technique is sharp per-pixel occlusion; however there are two significant disadvantages. First, the mismatch in the *fov* between the Creative Senz3D and the AR-Rift stereo cameras mean that occlusion can only be accurately applied to a small portion of the visible scene. Second, the Creative Senz3D makes the AR-Rift significantly heavier hindering the overall user experience. For these reasons, this approach was abandoned.

6.2.1.5.2 Skin segmentation and ray-casting

In this technique, skin color segmentation is applied to the images from the Logitech viewing cameras. Occlusion is determined by casting a ray from the viewing camera position to each fingertip. If the ray intersects no virtual objects, the background texture is displayed. This technique removes the need for the Creative Senz3D, however it is less accurate as occlusion is binary rather than based on depth, and is not computed per-pixel, i.e. when an object is intersected the occlusion is disabled. This can be overcome by calculating a disparity map from the stereo glasses, however this is a computationally expensive process (see Figure 6.5a).

6.2.1.5.3 Semi-transparent hand reconstruction

The final technique involved using the hand model that was constructed for computation of shadows as a semi-transparent proxy for the actual hand. This allowed users to see their actual hand with the virtual hand overlaid on top of it, while still having shadows cast on the virtual objects. The main advantage of this method compared to the others was that there was no additional hardware or computation required. An additional benefit was that virtual objects were still partially visible even when occluded so that the user could tell what was behind their hands (see Figure 6.5b). An additional calibration

step was required to align the virtual hands with the real hands as precisely as possible.

The “Skin Segmentation and Ray-Casting” and “Semi-Transparent Hand Reconstruction” techniques were evaluated during a pilot study, and participants reported that they felt they could perform the task well using either method. As the Semi-Transparent Hand Reconstruction technique required less computation it was chosen for occlusion in the experiment.

6.2.1.6 Seamless Object Creation and Interaction

G-SIAR supports dynamic creation of objects and every object can be interacted with using the G-Shell and G-Speech interaction methods. In the current iteration, object creation is supported using the “solid of revolution” method, where the user can draw the outline of an object and a solid model will be generated by rotating the outline around a central axis. Importing existing 3D models is also supported, as is model exporting, such that users can export their created models for 3D printing (See Figure 6.6).

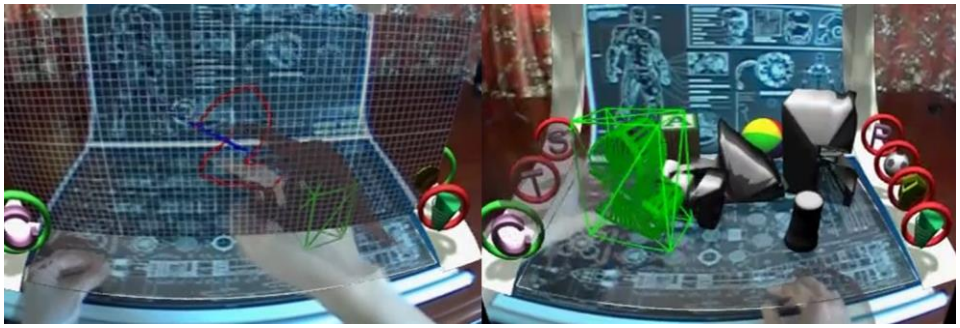


Figure 6.6: (a) User is drawing a figure “2” and (b) User is grasping the created object

6.3 Designing the Interaction Techniques

Within the G-SIAR framework, two interaction techniques were designed and implemented, Grasp-Shell and Gesture-Speech. The design approach was modular, so that either one or both techniques could be applied at any time.

6.3.1 Grasp-Shell (G-Shell)

A common assumption in AR is that virtual objects overlaid in the real environment should offer the same interactive affordances as real objects. From our user-defined gestures study (Piumsomboon et al., 2013) discussed in Chapter 4, we found that 39% of all the gestures elicited from participants for interaction in AR were "physical", meaning that they were gestures that acted physically on a virtual object as if it was a real object. As a result, our design goals for G-Shell were:

i. Demonstrate direct manipulation through natural hand gesture interaction.
ii. Learn from the user-defined gesture for AR study (Thammathip Piumsomboon et al., 2013) and apply the gestures elicited when possible.
iii. Support both interactivity-oriented and precision-oriented tasks.

6.3.1.1 Contacts, Shell and Penetration Distance

Natural physical interactions such as grasping require points of contact between the hand and object, however without any tactile feedback it is difficult for the user to know if their hands has made contact with a virtual object. Previous research (Benko et al., 2012; Hilliges et al., 2012) has applied constraints that disallow physical simulation hand proxies from penetrating an object. However,

in a physics-enabled simulation, the response to a collision between two objects is that one or both of the objects will move apart in a direction defined by the contact normal, a behavior referred to as “dynamic mode”. Because of this reaction, when attempting to grasp an object in a physically simulated environment, the user may unintentionally push the object away. To resolve this, we utilize a “kinematic mode”, where the contact response of an object is disabled so that the user has a full control over the object.

In order to provide a natural interaction technique that universally works for virtual objects of all shapes and sizes, Grasp-Shell (G-Shell) was developed. As the name suggests, G-Shell allows natural hand grasping of virtual objects using an invisible “Interaction Shell”, where contact with the shell changes the object from “dynamic mode” into “kinematic mode”, allowing for precise control and a standard gesture interface for objects, while supporting multiple types of physical gestures.

The interaction shell is essentially a collision shape that approximates the hull of the model. The simpler shape both reduces the less computation required for contact points and also yields better user performance when dealing with smaller objects as it makes them easier to grasp.

G-Shell requires that the interaction shell is always larger than the object’s collision shape, so that collision with the shell occurs before collision with the object. We define the “shell thickness” as the distance from shell surface to the objects collision surface, and the “Penetration Distance in Percentage of Shell Thickness (*PDST)*” as distance penetrated by the finger into the shell. Experimentally, we determined the optimal shell thickness for a tabletop setup to be 5mm.

By dividing the shell into multiple layers, we can offer both kinematic mode actions with the collision response disabled (no physical force applied to object) and dynamic mode actions where the collision response is enabled

(physical force will affect the object). G-Shell supports bimanual operation where two hands can be used at the same time to perform independent actions on separate objects. Listing 6.1 describes the G-Shell algorithm.

Listing 6.1: Simplified G-Shell Interface Handler Routine

```
1:  IF contact between finger(s) and shell(s) > 0
2:      Set each object to kinematic mode.
3:      IF object is not selected THEN select it ELSE deselect it.
4:      IF No thumb contact
5:          IF finger's PDST > 0% and ≤ 50% THEN
6:              Check for kinematic gestures
7:          ELSE IF finger's PDST > 50% and ≤ 120% THEN
8:              Object is being pushed
9:          ELSE IF finger's PDST > 120% THEN
10:             Check for kinematic gestures
11:     ELSE there is a thumb contact
12:         IF thumb and one of contacted fingers distance < threshold
13:             THEN Grasping = true
14:             ELSE Grasping = false
15:             Check PDST of each finger for other actions
16: IF Grasping on both hands is true and on the same object
17: THEN resize the object based on the change in distance
18:             between hands
19: ELSE
20:     Move object(s) on each hand that Grasping is true
21:     IF fling condition is met THEN
22:         Disable kinematic mode and apply
23:         impulse, J = m * Δv = Δp on the object(s)
```

6.

Figure 6.7 shows a shell cross section of a Porsche model. We define a "safe zone" as a *PDST* greater than 0% but less than 50%. This safe zone is intended

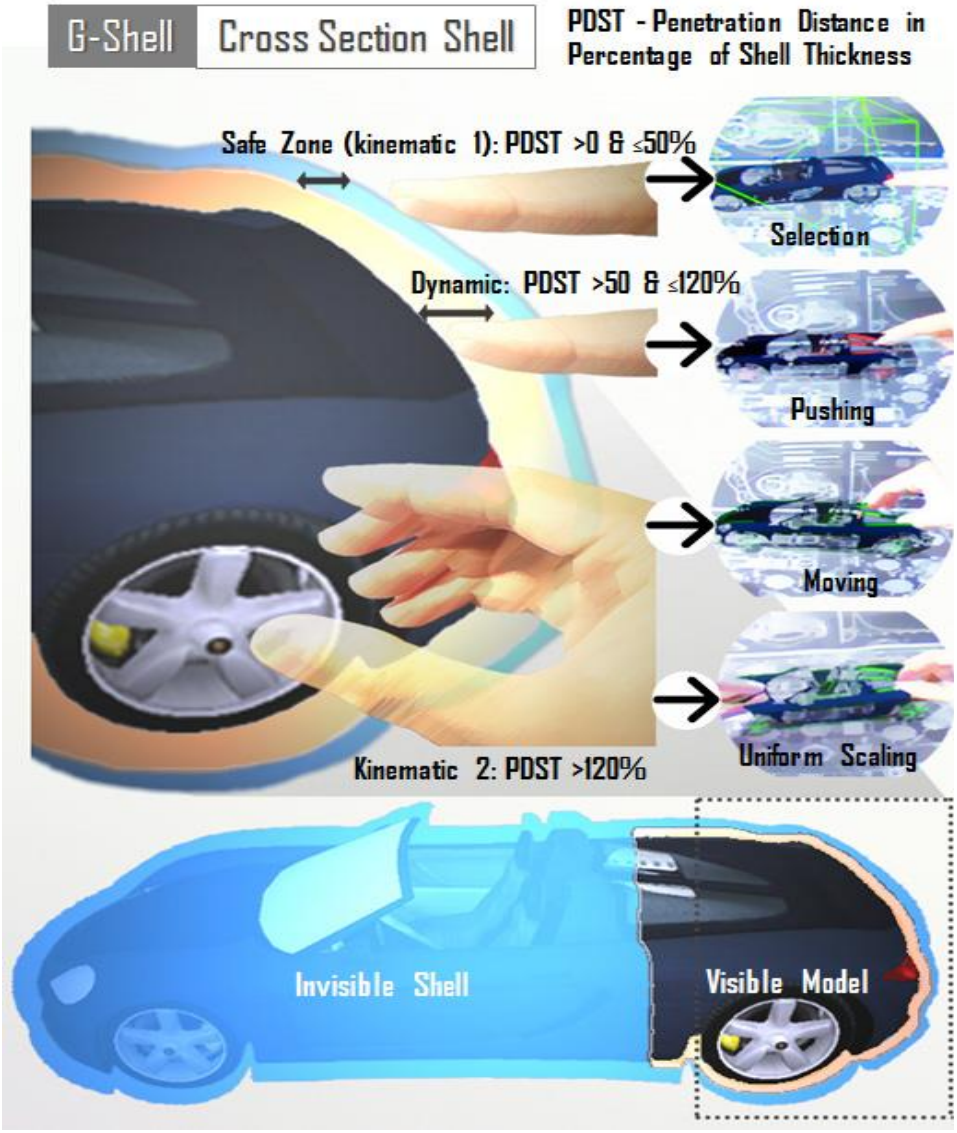


Figure 6.7: G-Shell's cross section and visualization

for kinematic mode and object selection, and when activated the model of the object turns semi-transparent and a red outline is shown around the object to indicate that kinematic mode is activated. In kinematic mode, collision response is disabled and touching an object will toggle its "selected" property. This object selection allows for non-physical manipulations, such as changing the color of an object.

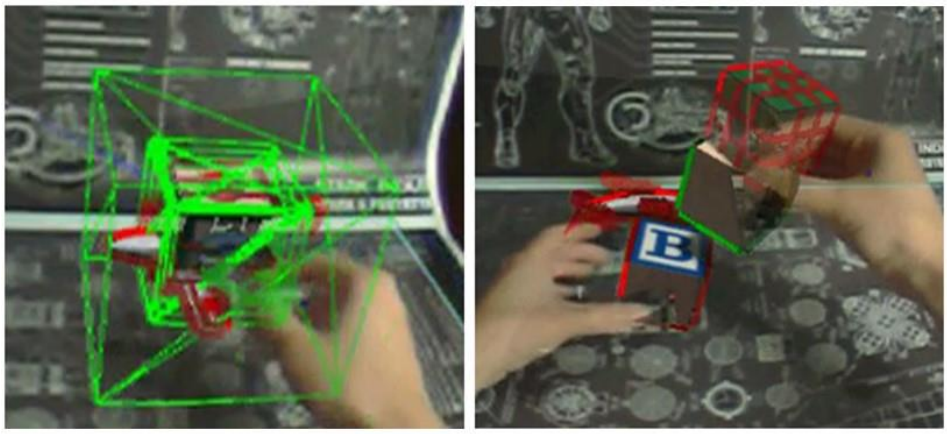


Figure 6.8: G-Speech's move (left) and G-Shell's move (right)

At a *PDST* beyond 120%, the object is also in kinematic mode. A single handed grasp action in this zone would cause the object to be transformed (see Figure 6.8 (right)). A grasp is defined as occurring when the distance between the thumb and the opposing finger that made the contact is smaller than a threshold of 20mm. In the event of a successful grasp, the object's outline turns from red to green. The relative transformation between the object and the hand at time of contact is stored, and any translation or rotation of the hand is applied to this transform. The grasping transformation is a 6 *dof* manipulation, such that the hand can translate and rotate the object at the same time. This transformation also supports for multiple objects simultaneously, as shown in Figure 6.8 (right).

A bimanual grasp, i.e. grasping with a single object with both hands, allows for uniform object scaling (see Figure 6.9). The change in size was determined by the difference between the position of the thumbs at the start time of the grasp and the current position. At the end of the resizing, the user confirms the change by releasing the grasp on their right hand first, or cancels the change by releasing the grasp on the left hand first.

6.3.1.3 Actions in Dynamic Mode

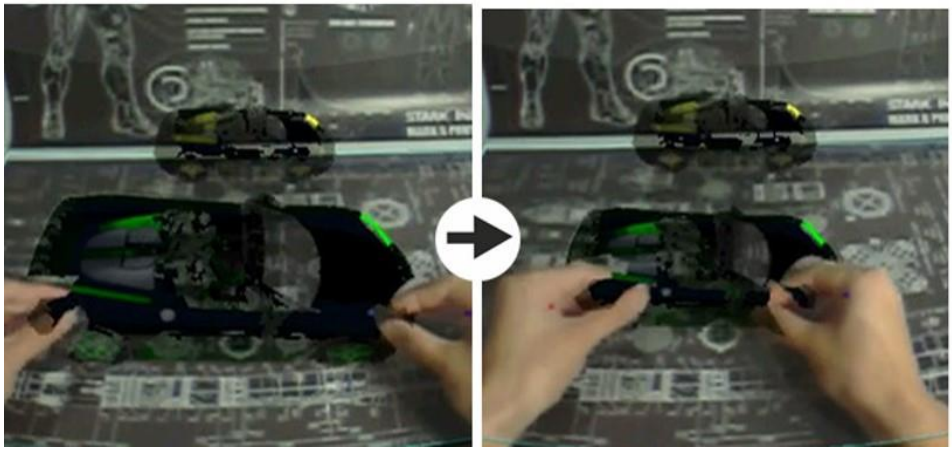


Figure 6.9: G-Shell's resize

Two dynamic actions are supported in G-Shell, pushing and flinging. Pushing is enabled at a *PDST* between 50% and 120%. By allowing penetration up to 120% collision between a fingertip and object collision shape is guaranteed, causing the object to move a small distance in response. During pushing, any contact made with the thumb disables the dynamic mode and enables kinematic mode, as all gestures involving the thumb are kinematic.

To fling an object, the user must first grasp the object, then quickly release it while moving their hand in the direction of the fling. A fling is only executed when the velocity between the thumb and index finger exceed a given threshold of 0.3m/s. The impulse applied to the flung object is taken as the mass of the object times the change in hand velocity, which is equivalent to the change in momentum of the object. The velocity vector is calculated as the difference between the initial and final position of the hand.

6.3.2 Gesture-Speech (G-Speech)

G-Speech is a multimodal gesture-speech interaction that offers deictic gesture and metaphoric gestures. The design recommendation from the Wizard of Oz (WOz) study (Lee & Billinghurst, 2008) was used in the design and implementation of G-Speech. A Woz study which measured speech and gesture timing in a multimodal interface in AR, it was found that 65% of all the gestures used were deictic, followed by 35% of all gestures being metaphoric. Based on this the authors provide the following design guidelines: use an accurate gesture-triggered system that is adept at recognizing pointing and metaphoric gestures, use context-based multi-signal fusion, use phrase-based speech commands, provide audiovisual feedback, and learning modules should be applied in the multimodal fusion architecture.

Based on this our design goals for G-Speech were:

i. Demonstrate an indirect manipulation through hand gesture and speech.
ii. Use the design recommendation from the WOz study (Minkyung Lee & Billinghurst, 2008).
iii. Support fast detection and interpretation of deictic gesture.
iv. Apply suitable metaphoric gestures learned from user-defined gesture for AR study (Thammathip Piumsomboon et al., 2013).

6.3.2.1 Deictic Gestures

As described in Section 6.2.1.2, fast pointing detection can be calculated using the hand's geometry. Pointing is restricted to the index finger or index and middle fingers only. By comparing the distance between the index, ring and pinky fingertips to a stable reference point such as the wrist, a probability that the hand is pointing can be calculated. Visual feedback that a pointing was

recognized was provided as a ray cast from the index finger into the distance, which was a red color for the left hand and blue for the right (see Figure 6.10 (right)).

When this pointing ray intersected an object, the ray ended at the point of intersection and turned green. A yellow box would then appear around the object, and the user had a choice to utter a command to perform an action such as “move it”. Objects that were being pointed at could be selected by saying “select”, which would be acknowledged as the yellow bounding box turning green. Multiple objects could be selected consecutively, otherwise “select all” could be used to select all virtual objects in the scene. To deselect, the user could point at an object and say “deselect”, or saying “cancel selection” would deselect everything.

6.3.2.2 Intuitive and Indirect

In the real world, multi-modal gesture and speech commands are often very indirect, and make use of understanding of real world geometry and physics. For example, when issuing a command to move an object, one might point at a place and say “move it there”. One would expect that the object would be placed

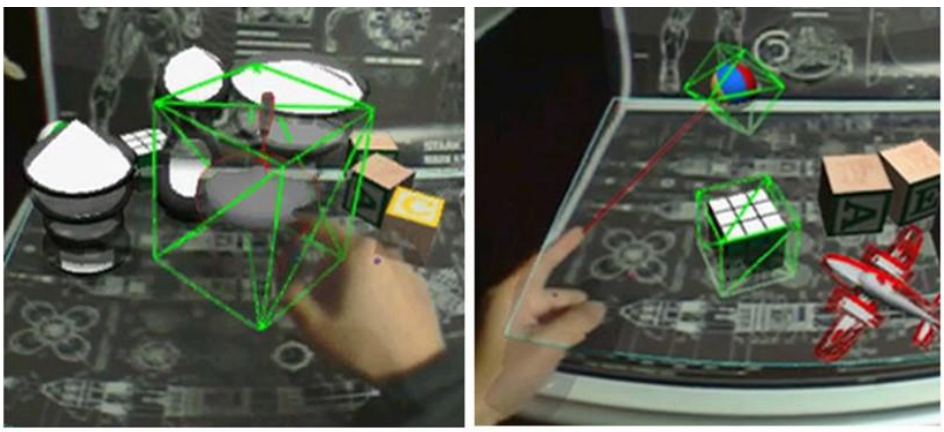


Figure 6.10: G-Shell’s selection (left) and G-Speech’s section (right)

in an appropriate location for the type of object, for example a placing a flower pot on a table or a hanging a picture on a wall.

In a virtual 3D environment, these limitations and assumptions may not apply; the target location might be floating in 3D space where there is nothing for the pointing ray to intersect with. Consequently, for this study G-Speech was designed to support continuous actions such that when the user pointed at an object and said "move it", the object would be attached to the ray extending from the user's index finger (see Figure 6.10 (right)). The user could move the object and say "release" to confirm the movement or "cancel" to cancel the action. The user could issue additional commands such as "rotate" without saying "release" and the change would also be confirmed.

When observing how people interacted indirectly using gesture and speech in the real world, it was noticed that translation and rotation operations were usually separate, for example saying "move the table there" while pointing at the location on the floor, followed by "rotate it by this much" and showing the amount of rotation by twisting their hand. For this reason, it was decided to separate translation and rotation operations in G-Speech, as opposed to the 6 *dof* interaction offered in G-Shell.

6.3.2.3 Metaphoric Gesture

The gestures for rotation and uniform scaling were designed based on metaphors observed in the real world. For rotation, the user would say "turn it" and the amount of rotation was indicated by a change in orientation of the user's hand. Rotation could also be performed for multiple objects at the same time as shown in Figure 6.11.

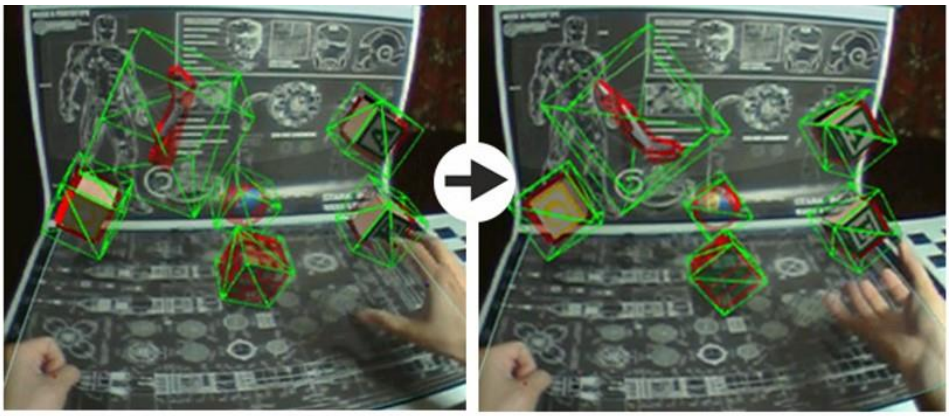


Figure 6.11: G-Speech's rotation

In case of scaling, the metaphor used was describing the object size using the distance between the two hands or the thumb and index finger of one hand. In our implementation the scale factor was determined by the difference between the current and initial distances between the hands. The user was given continuous visual feedback showing the change in rotation and scale (See Figure 6.12).

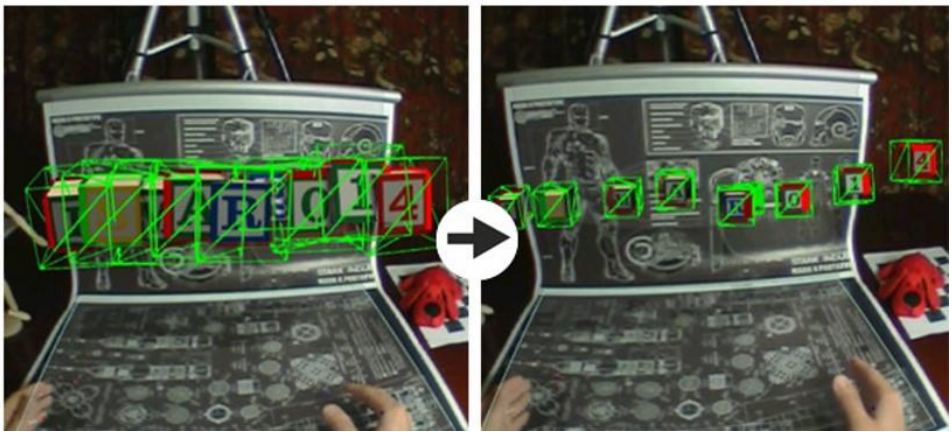


Figure 6.12: G-Speech's resize

6.3.2.4 Gesture and Command Pair

Although G-SIAR supports unlimited pairing of gestures and commands, for the user study the number of commands was limited to the nine shown in Listing 6.2. This list was determined in a pilot study as a sufficient number of commands to complete the proposed tasks, while not requiring significant learning time or overloading the user's memory.

Listing 6.2: List of commands used in the experiment.

1: **SELECT**	6: **RELEASE**
2: **DESELECT**	7: **CANCEL**
3: **MOVE IT / TRANSLATE**	8: **SELECT ALL**
4: **TURN IT / ROTATE / TWIST**	9: **CANCEL SELECTION**
5: **RESIZE**	

6.4 Discussion

6.4.1 Performance

On the Alienware 17 laptop (Intel Core i7-4800MQ @ 2.70Ghz with Nvidia GeForce GTX 780M), G-SIAR could update the display at over 50 FPS while tracking and interaction states were updated at more than 25 FPS. During the pilot studies it was found that experienced users could easily move, rotate, and scale virtual objects to match a target with errors of less than 5mm, 5° and 5%, respectively.

6.4.2 Limitations

There are a number of limitations in the G-SIAR framework as follows:

6.4.2.1 Use of an In-place Depth Sensor

To track the user's hands, a depth sensor is placed above the interaction space facing downward, limiting the interaction space to a defined volume between the table and the sensor. To give the user more freedom, the sensor could be mounted onto the HMD so that the interaction space is always in front of the user.

6.4.2.2 Use of an Image Marker

In this research, we utilize an image marker to provide robust head tracking and to synchronize the coordinate systems for the depth and viewing cameras. However, to make the system robust to occlusion of the marker from the hands and more extensible, the image marker could be replaced by tracking features that exist naturally in the environment.

6.4.2.3 Limited Number of Gestures Supported

For the experiment, we only chose to support a subset of available gestures to not overload the user. However, G-Shell can support a wider range of gestures including all the physical gestures elicited from the user-defined gestures study in Chapter 4. We plan to support more gestures and validate these gestures in future work.

6.5 Lessons Learnt and Research Questions Raised

A number of lessons were learnt during the design and development of the G-SIAR framework and G-Shell and G-Speech interaction techniques, which are summarized in this section.

6.5.1 Architecture of Multimodal AR Framework

G-SIAR was designed to encapsulate the core components of a Multimodal AR platform, as defined in Section 3.4.2. The first component, tracking to determine the transformation of the user viewpoint, was implemented using image-based marker tracking. The second and third components, reconstruction, recognition and tracking, were implemented using a depth sensor placed above the interaction area to capture the environment and user's hands. The forth component, collision detection and physics simulation is a core part of the G-Shell interaction technique. The final component, feedback, is provided as both visual, in the form of hand occlusion, shadows, outlines, etc, and audio, in the form of sound effects during interaction, for example a sound indicating contact between hands and objects.

The successful implementation of the G-SIAR framework based on these components confirms our finding in Chapter 3.

6.5.2 Improve Interaction with Translucent Hands

During the development of G-SIAR, pilot studies were conducted to compare user preference of hand occlusion between opaque and translucent hands. Pilot study subjects reported that they could perform well using either technique. However, the latter technique is more computationally efficient and was chosen for the study reported in the next chapter.

6.5.3 Direct Manipulation with Natural Hand Interaction

With the G-Shell interaction technique we introduced the concept of an interaction shell where the penetration distance of fingers into different layers of the Shell provided different modes of interaction. Two modes were introduced, kinematic and dynamic, where the former gives the user gestural

control of the targeted object, and the latter applies physical forces to the object. Through the use of the shell, direct manipulation with natural hand interaction is not restricted to only physical interaction but it can support gestures for more complex tasks with higher precision. A usability evaluation of this interaction technique is presented in Chapter 7.

6.5.4 Indirect Manipulation with Gesture and Speech

G-speech was implemented following best practice and guidelines found in previous research. The gesture that are critical for multimodal gesture and speech interface are deictic gestures and metaphoric gestures. Deictic gestures use hand pointing to provide spatial context and speech to issue commands. , while for metaphoric gestures, the objective depends on the context of the interaction. For example, in G-SIAR the rotation task uses a hand gesture to determine the amount of rotation, while the scaling task uses a different gesture to determine the new size of the object.

In Chapter 7 we present a usability study for the G-Speech technique, and show that the combination of deictic and metaphoric gestures is effective and intuitive.

6.6 Conclusion

This chapter describes the design and development process for a new multimodal AR framework, G-SIAR (Gesture-Speech Interface for Augmented Reality). G-SIAR was implemented following the guidelines laid out in earlier chapters. The framework provides an immersive and large interaction space, physics simulation with high accuracy, real-time and seamless object creation and interaction, and realistic rendering with shadows and hand occlusion. It offers natural hand interaction and gesture-speech interaction as native inputs.

Two interaction techniques were implemented in the G-SIAR framework: Grasp-Shell (G-Shell), a direct natural hand interaction technique and Gesture-Speech (G-Speech), an indirect multimodal gesture-speech interaction technique. The motivations for various design decisions for each interaction technique were explained, and the performance and limitations of the framework and techniques were presented.

In the following Chapter, we test and compare these two interaction techniques in term of usability, performance, and preference. More importantly, the study can help identify the strengths and weaknesses of each interaction technique so that appropriate guidelines for improving each technique can be attained.

Chapter 7
Comparison Study between Direct and Indirect Natural Interaction in Augmented Reality

In the previous chapter, two natural interaction techniques were presented: *Grasp-Shell* (G-Shell), a novel direct natural hand interaction technique, and *Gesture-Speech* (G-Speech), an indirect multimodal interaction technique. This chapter presents a user study that evaluates these two techniques, where users were required to perform tasks involving single object transformation, multiple object transformation, and uniform scaling. From the results, differences were found between the interaction techniques in regards to efficiency, usability, and preference. This research examines the impact of directness and indirectness of interaction in each task, and the implications of these results.

This chapter's primary contributions are: (1) Results from a formal user study comparing the two interaction techniques discussed in the previous chapter and (2) Discussion on the findings and their implications.

7.1 User Study

This within subjects study focuses on the usability and user impressions of the two interaction techniques, G-Shell and G-Speech. We believe that each technique has merits and weaknesses, and that neither technique is universally better as different levels of directness of interaction will be better suited to different tasks. We predict that relocation tasks will benefit from the direct interaction of the G-Shell technique, as it allows users to apply their real-world experience in grasping and moving the object(s). In contrast, resizing an object

may be more challenging using direct manipulation due to the fine control of both hands required when resizing small objects, and as such may be easier completed with G-Speech.

With this reasoning, we propose the following hypotheses:

(*H1*) There is a difference in the resulting performance, usability, and preference between G-Shell and G-Speech when relocating object(s) in 3D space.

(*H2*) There is a difference in the resulting performance, usability, and preference between G-Shell and G-Speech when uniform resizing an object.

7.1.1 Experiment

Our user study was designed so that all virtual objects would be in an arm-reachable near-space environment with dimensions of 600mm wide, 450mm deep, and 450mm high, and all virtual objects would range in size from 30mm to 400mm. The experiment was comprised of three tasks; (1) *single object relocation*, (2) *multiple object relocation* and (3) *uniform scaling*. The decision was made to focus on these tasks as they encapsulated fundamental actions that are the basis of many other functions, e.g. touching, grasping, moving and releasing for G-Shell as well as pointing, moving, uttering commands for G-Speech, while maintaining a task load that was appropriate to complete within a single session of the experiment.

7.1.1.1 Apparatus and Experimental Setup

The equipment used in this experiment is as follows:

A. AR-Rift (See Figure 7.1a)
B. PrimeSense Carmine 1.09 and a tripod (depth sensor)
C. Creative Senz3D camera (stereo microphone)
D. Image-based marker (A1 paper size)

E. Alienware 17 laptop (Intel Core i7-4800MQ @ 2.70Ghz with Nvidia GeForce GTX 780M)

F. Fluorescent lighting panel

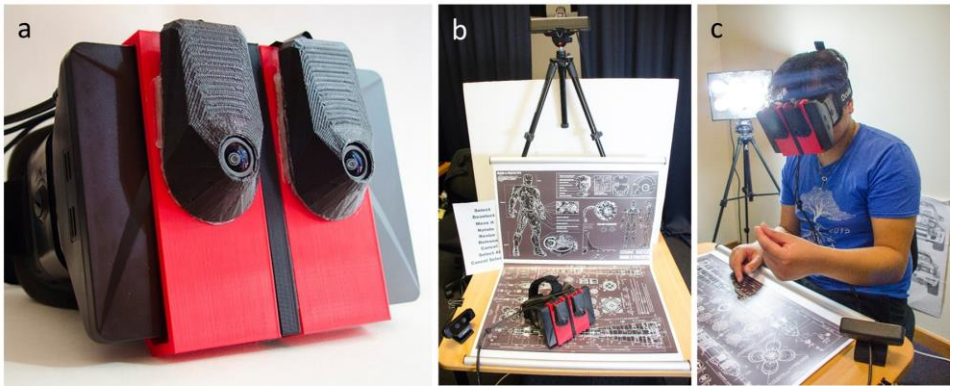


Figure 7.1: (a) AR-Rift, (b) Experimental setup, (c) A subject interacting with the system

The experimental setup is as shown in Figure 7.1b and 7.1c. The participant sat in front of the image-based marker wearing the AR-Rift. The stereo cameras mounted on the AR-Rift used the image marker to determine the head pose. A PrimeSense sensor was placed on a tripod above the table pointing downward. The microphone and a list of commands for G-Speech, printed on an A4 paper, was placed on the table to the user's left hand side. A lighting panel was placed beside the participant, as shown in Figure 7.1c, to ensure adequate illumination for the computer vision algorithm.

7.1.1.2 Experimental Conditions and Measurement

Chapter 6 describes two interaction techniques, G-Shell and G-Speech. These two interaction techniques are the independent variable in this experiment. The response variables are; (1) task completion time, (2) NASA TLX, which includes the five dimensions; effort, frustration, mental, physical and temporal demand, (3) a 7-points Likert scale usability rating, which includes eight factors:

learnability, efficiency, memorability, accuracy, satisfaction, intuitiveness, naturalness, and fun, and (4) preference, where the participants must vote for their preferred condition for each task. Further details on data collection and analysis are explained in Section 7.1.1.3.

7.1.1.3 Data Collection and Analysis

The data recorded during the experiment included the task completion time for each subtask, the discontinuity occurrence of G-Shell (number of times grasped and released), the usage of each command for G-Speech and the distance the hands travelled for both interaction techniques. The questionnaire included a 7-point Likert scale usability rating and NASA TLX for each condition, and asked the user's preference for each task. Videos of the experiment from the participant's point of view were recorded throughout the experiment.

Task completion time (**tct)**, usability rating, and NASA TLX were compared between G-Shell and G-Speech for each task. Each task was analyzed independently and a paired T-test for **tct** was calculated. A Wilcoxon signed-rank test with continuity correction was applied on the usability rating, and NASA TLX result.

7.1.1.4 AR Interface Supported under Experimental Condition

In the experiment, both conditions shared the same AR interface provided by G-SIAR framework, and there was no difference in the interface for each condition. Common features supported by AR interface include: an AR view transformed according to the user's head position, hand tracking and occlusion, and audio feedback during contacts for G-Shell or selections for G-Speech.

7.1.1.5 Performing the Interaction

A detailed description of both interaction techniques is provided in Section 6.3. In this section, we briefly describe the actions that the participant was required to perform for each condition. The interaction performed in this experiment consisted of translation, rotation, and scaling of virtual objects.

For G-Shell, translation and rotation were combined into a single 6 *dof* hand movement. The participant could grasp the targeted object with a single hand, either left or right using their thumb and at least one opposing finger. Once the outline of the object turned from red to green, it indicated that the object was movable.

For G-Speech, the participant could point at the object and use their voice to issue a command. A list of the nine commands available for G-Speech, as determined in a pilot study, is shown in Listing 7.1. To translate the user would point to the object and issue the "select" command to select the object. After selecting the object, the user would issue a "move it" command and targeted object would attach to the participant's hand that was used for pointing. The object could be moved to the desired location and it could be released with the command "release". For rotation, once the "turn it" speech command had been executed, the targeted object would remain in-place and rotate about its center according to the rotation of the hand.

Scaling with G-Shell was performed be grasping the object with two hands and moving them closer together or further apart. For G-Speech, once the resize command was executed, the object's size would change depending on the relative distance between the two hands. For both condition, moving hands apart would enlarge the object and moving hands together would shrink it.

Listing 7.1: List of commands used in the experiment.

1: **SELECT**	6: **RELEASE**
2: **DESELECT**	7: **CANCEL**
3: **MOVE IT / TRANSLATE**	8: **SELECT ALL**
4: **TURN IT / ROTATE / TWIST**	9: **CANCEL SELECTION**
5: **RESIZE**	

7.1.1.6 Participants

Twenty one participants were recruited for the study, eleven males and ten females, with an average age of 24.9 (SD = 6.09) years. Seventeen of the participants were right handed, two were left-handed and two could use both hands equally well. Nine participants had some experience with AR but only one considered himself knowledgeable on the subject. Twelve had some experience with gesture interfaces from playing with Kinect or Wii. Twelve had some experience with speech interfaces from using Siri or Xbox. There were nine native, seven fluent and five intermediate English speakers. All of the participants could communicate well and understood the experimental protocol.

7.1.1.7 Tasks

Task 1 focused on relocation of single object, and featured ten subtasks, involving both translation and rotation (See Figure 7.2 (left)). Task 2 focused on relocation of multiple objects and was comprised of five subtasks involving translation, and translation and rotation (See Figure 7.2 (right)). Task 3 focused on resizing objects of varying shapes and sizes and was comprised of five subtasks (See Figure 7.3).

For Task 1 and 2, the object size ranged from 60mm to 100mm. Users had to transform the object to match a target object, such that the position and orientation difference between the object and the target must be smaller than 20mm and 15°, respectively. For Task 3, the requirement was that the object's scale must be within 10% of the target's scale. These limits were determined from two pilot studies, where the matching requirement values were varied between 10, 15, and 20mm for position, 5°, 10°, and 15° for orientation, and 6%, 8%, and 10% for scaling. During these pilot studies, the average time taken by users with no prior experience with either interaction techniques were

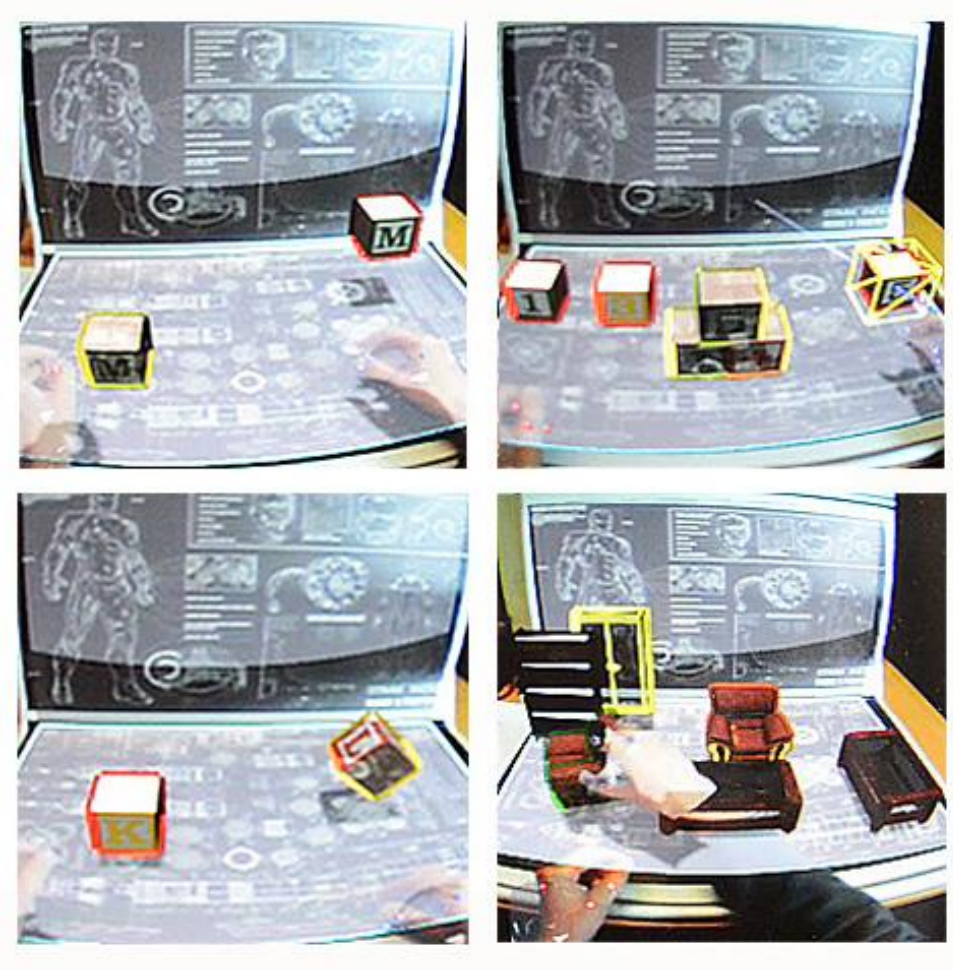


Figure 7.2: Object relocation in task 1 and 2.

measured, and the final values were decided upon based on time taken. During the pilot studies it was also discovered that experienced users could easily achieve matching conditions of 5mm, 5° and 5% for position, rotation and scale respectively.

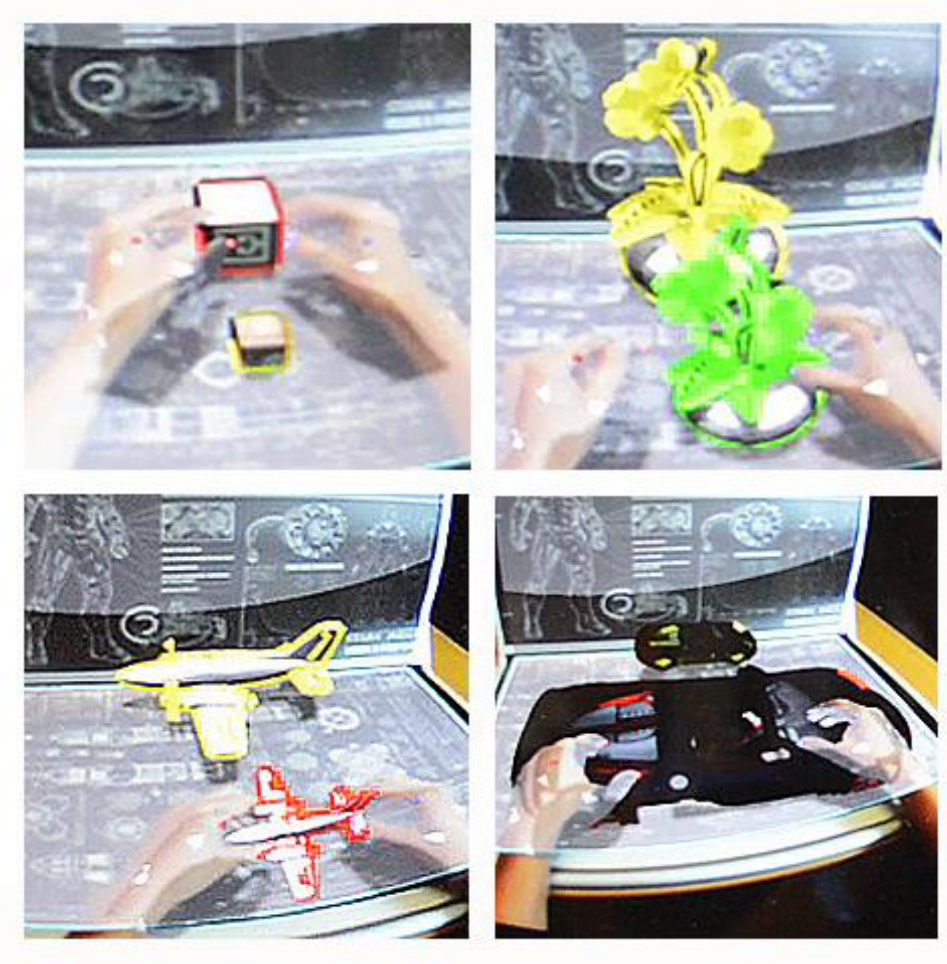


Figure 7.3: Uniform resizing of various objects in task 3.

7.1.1.8 Procedure

To counterbalance conditions, the presentation order of the tasks was randomized and participants were equally distributed based on their gender.

The experiment took 1 to 1.5 hours to complete. Before commencing the experiment, each participant was given 5 minutes per interaction to learn in a sandbox program and another 5 minutes in a practice session that was similar to the experiment. During this learning period, the system was calibrated to ensure optimal accuracy for each participant. This calibration involved the adjustment of parameters to ensure accurate gesture recognition regardless of hand size, and selection of verbal commands that the speech recognizer could accurately determine for each participant.

At the beginning of each task, the participant had a 3 second countdown, which displayed in the center of their view. As each task began and for every successful target matched, a sound of a bell was played. The object was displayed with opaque textures and a red outline, while the target was a 50% translucent render of the object with a yellow outline (See Figure 8.1). The task completion time started after the bell rang and stopped when all the targets in the scene were matched.

7.1.2 Results

The following sections present the results for the comparison of the two interaction techniques with regards to **tct**, NASA TLX, 7-point Likert scale usability rating, user preference, and general user feedback.

7.1.2.1 Task Completion Time (tct)

The G-Shell technique was significantly faster than G-Speech for single and multiple object manipulation, but not for scaling (See Figure 7.4). The T-test showed a significant difference between the two interaction techniques in term of **tct** for Task 1 *single object relocation*, with M = 23.77s (SD = 25.72) for G-Shell and M= 42.69s (SD = 73.5) for G-Speech where $t(209) = -3.78$, $p < 0.001$.

The same was true for Task 2, *multiple object relocation*, with M = 56.94s (SD = 58.03) for G-Shell and M = 103.17s (SD = 87.2) where $t(94) = -6.91$, $p < 0.001$. For Task 3, there was no significant difference in **tct** where M = 16.0 (SD = 22.57) for G-Shell and M = 12.21 (SD = 10.17) for G-Speech.

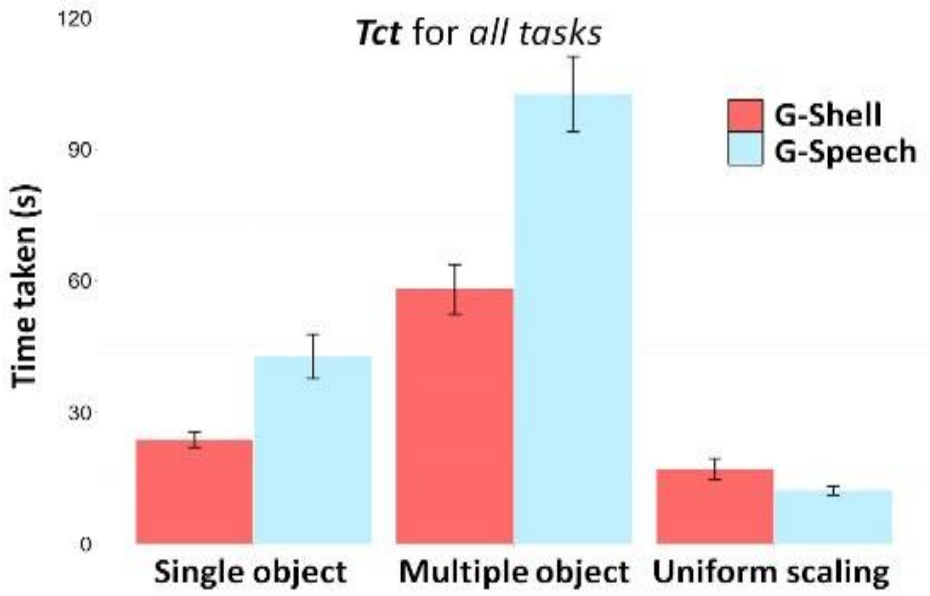


Figure 7.4: Task completion time, error bars represent +/-SEM.

7.1.2.2 NASA TLX

G-Shell required significantly less *effort*, *frustration*, *mental*, *physical* and *temporal demand*, and provided significantly higher *performance* for Task 1. G-Shell required significantly less *temporal demand* for Task 2. G-Speech was significantly less *frustrating* for Task 3 (See Figure 7.5).

A Wilcoxon signed-rank test with continuity correction yielded a significant difference between the two interaction techniques in every category for Task 1, which comprised of *mental demand* ($V = 37.5$, $p = 0.012$), *physical demand* ($V = 22$, $p = 0.033$), *temporal demand* ($V = 8.5$, $p = 0.01$), *performance* ($V = 104$,

p = 0.001), *effort* (V = 38.5, p = 0.013), and *frustration* (V = 19, p = 0.012). For this task, G-Shell required lower *mental demand*, *physical demand*, *temporal demand*, *effort*, *frustration*, and higher *performance*. However, a significant difference could only be found for *temporal demand* in Task 2 (V = 20, p = 0.043), which favored G-Shell, and *frustration* in Task 3 (V = 66.5, p = 0.034) that favored G-Speech.

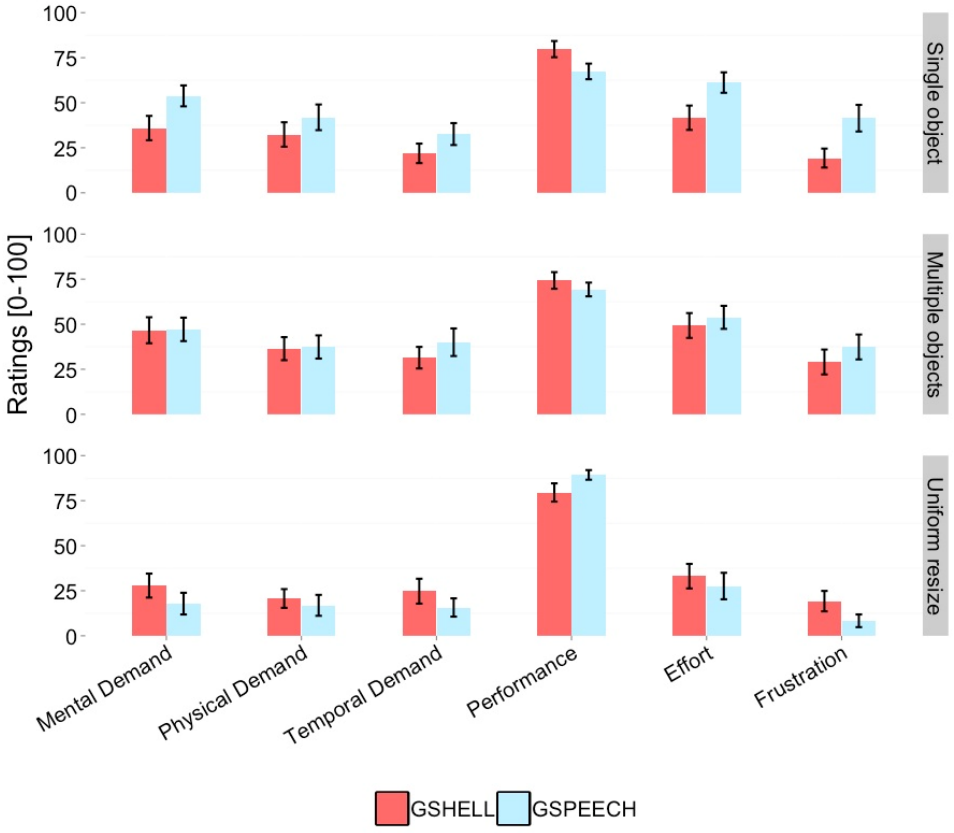


Figure 7.5: NASA TLX, error bars represent +/-SEM.

The same test was applied to compare Task 1 and 2 for each interaction. The test showed a significant difference between Task 1 and 2 for G-Shell in two categories, they were *mental demand* (V = 29, p = 0.014) and *temporal demand* (V = 13.5, p = 0.027).

7.1.2.3 Usability Ratings

G-Shell was rated significantly better for *single object relocation* and G-Speech was rated better for uniform resizing. There was no significant difference in the *multiple object relocation* task (See Figure 7.6).

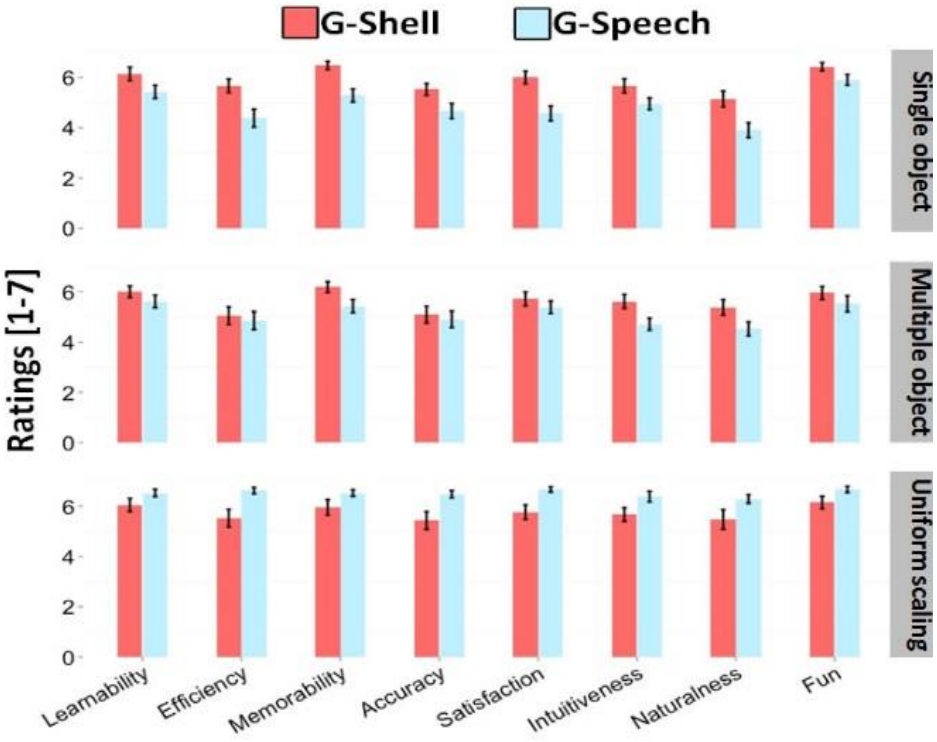


Figure 7.6: Usability rating, error bars represent +/-SEM.

A Wilcoxon signed-rank test with continuity correction was applied for usability ratings and resulted in a significant difference between the two interaction techniques in every attribute for Task 1, with G-Shell showing higher ratings for *learnability* ($V = 87$, $p = 0.029$), *efficiency* ($V = 138$, $p = 0.003$), *memorability* ($V = 110.5$, $p = 0.004$), *accuracy* ($V = 96$, $p = 0.041$), *satisfaction* ($V = 144$, $p = 0.001$), *intuitiveness* ($V = 98$, $p = 0.028$), *naturalness* ($V = 123$, $p = 0.004$), and *fun* ($V = 64.5$, $p = 0.042$).

For Task 2, only *memorability* (V = 95, p = 0.006), and *intuitiveness* (V = 114, p = 0.016) were significant in favor of G-Shell. For Task 3, *efficiency* (V = 14, p = 0.016), *accuracy* (V = 16, p = 0.023), *satisfaction* (V = 13.5, p = 0.013), and *intuitiveness* (V = 4, p = 0.009) were rated significantly higher for G-Speech.

The *goodness* score for G-Shell and G-Speech, was taken as the average of all ratings for each task. The ratings for Task 1 were 5.88 and 4.89, for Task 2 were 5.63 and 5.12, and for Task 3 were 5.75 and 6.52, for G-Shell and G-Speech respectively. There was a significant difference between the two interaction techniques for Task 1 in favor of G-Shell (V = 211.5, p < 0.001) and Task 3 in favor of G-Speech (V = 32, p = 0.037).

7.1.2.4 Preference

Figure 7.7 shows that G-Shell was more preferable for *single and multiple object relocation* tasks, while G-Speech was preferred for the uniform resizing task. It was found that participants with prior gesture interface experience performed significantly better than those without in terms of **tct**. The votes for each task

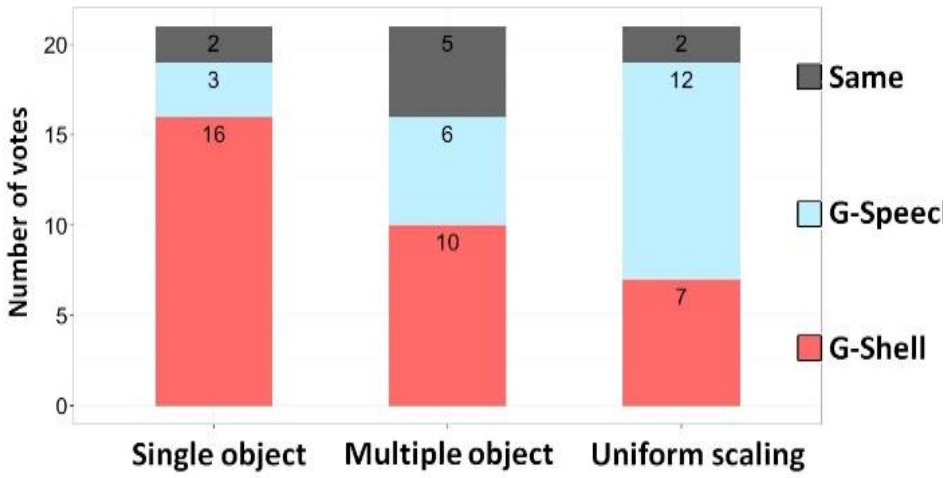


Figure 7.7: Preferences for all tasks.

were cross-referenced and it was found that for Task 2 *multiple object relocation*, the preference of G-Shell, G-Speech, and neither was 6 votes, 1 vote, and 4 votes respectively for users with gesture interface experience, and 4 votes, 5 votes, and 1 vote respectively for those without.

7.1.2.5 General Feedback

The subjective feedback provided by the users was analyzed, and common themes were found. These themes are summarized into 7 motifs as follows:

Challenging but enjoyable and fun. Some participants found that performing certain tasks with certain interaction techniques was challenging. Nevertheless, all participants described the experiment as enjoyable and fun. For example, after performing the *multiple object relocation* with G-Speech subject *P2* commented, *"It was challenging a bit, which was part of the fun but can also be irritating sometimes".*

Natural and intuitive to grasp. Most of the participants found G-Shell's 6 *dof* move and bimanual scaling, natural and intuitive. Subject *P13* gave the following comment, *"Love it. The interface is natural and requires little extra thought to manipulate. There is a very high level of similarity to reality, with the hand movements being intuitive".*

Difficult imagining an indirect rotation. Some participants found it hard to conceptualize rotation remotely with G-Speech. *P9* expressed *"the rotation, again, was hard to imagine"* and *P11* commented, *"I like this interface but would like more practice with rotation".*

Easy and intuitive to resize. All of the participants found G-Speech's scaling interaction, easy and intuitive to use. Subject *P12* said, *"Very easy to use, intuitive, and allows for very accurate object manipulation, very efficient, and looks to be easy to use on multiple objects. Can think of no drawbacks! ".*

Smaller is harder. Some participants found it challenging to directly resize a small object with G-Shell. *P14* commented, *"Getting the finger in the right place for both hands is tricky"*.

Better control and so more precision. The introduction of the 6 *dof* move in G-Shell gave more freedom but reduced precision when the user wished to change only the position or rotation. With regards to G-Speech, *P3* stated, *"I liked this interface better because I felt like I had more control over the manipulations when I had to rotate or move the object with my hands. Mostly, this was because the objects also responded to my voice, so I didn't need to worry about not having perfect fine motor skills in this program".*

To speak or not to speak. The preference of interaction varied between participants and one of the key factors for this was the ability to use speech. *P4* favored G-Shell for every task stating that, *"I don't like to keep talking all the time during the tasks"*, while *P7,* who favored G-Speech for every task stated that, *"Because actions without the voice commands takes more effort, It was easier with voice commands to an extent".*

7.2 Discussion

In the following sub sections we discuss some of the findings from this evaluation.

7.2.1 G-Shell vs G-Speech

The differences that were found in the statistical analysis of the two interaction techniques can be summarized in six main points:

7.2.1.1 Our Hypotheses

Our hypotheses for this experiment were as follows:

(*H1*) There is a difference in the resulting performance, usability, and preference between G-Shell and G-Speech when relocating object(s) in 3D space.
(*H2*) There is a difference in the resulting performance, usability, and preference between G-Shell and G-Speech when uniform resizing an object.

We found support for both hypotheses as differences in performance, usability, and preference when performing each task.

We found that G-Shell was significantly faster than G-Speech for relocating object(s), but not for uniform resizing of an object. G-Shell required significantly less *effort, frustration, mental, physical* and *temporal demand*, and provided significantly higher *performance* for single object relocation. G-Shell required significantly less *temporal demand* for multiple objects relocation. G-Speech was significantly less *frustrating* for uniform resizing. G-Shell was rated significantly better for single object relocation and G-Speech was rated better for uniform resizing. G-Shell was more preferable for single and multiple object relocation tasks, while G-Speech was preferred for the uniform resizing.

7.2.1.2 Translation and Rotation Task

G-Shell was more efficient on average for combined translation and rotation tasks. It is possible that the main reason for this is due to G-Shell supporting 6 *dof* interaction, such that a single action can position and orient an object as opposed to two separate actions as required by G-Speech.

7.2.1.3 Single Object Relocation

For single object relocation, G-Shell required less effort, frustration, mental, physical and temporal demand, and provided higher performance. However, for multiple object relocation G-Shell only required less temporal demand and for uniform resizing G-Speech was less frustrating. The results for temporal

demand were consistent with the time taken for single and multiple object relocation where G-Shell took less ***tct*** on average. Based on video analysis, we believe that the higher frustration with G-Shell for uniform scaling task may be due to the difficulties of directly manipulating objects of varying shape and size.

7.2.1.4 Task Load Index

In terms of task load index, participants perceived that the multiple object relocation task was more mentally and temporally demanding than the single object relocation task when using G-Shell. Introducing multiple objects has an impact on the G-Shell interaction technique as it requires direct object contact and hence an increase in the number of objects increases the number of actions that must be performed. This may explain the significant rise in perceived *mental* and *temporal demand* in the two tasks for G-Shell.

7.2.1.5 Usability Rating

In terms of usability, both interaction techniques were rated positively in all categories in every task, while G-Shell was rated significantly better for single object relocation and G-Speech was rated better for uniform resizing. Although, G-Shell was rated higher on average than G-Speech in the multiple object relocation task, there was not a significant difference between them.

7.2.1.6 User Preference

In term of preference, the majority of participants voted for G-Shell for single and multiple object relocation tasks, while G-Speech had the majority for uniform resizing task. This matches our expectation that neither technique would be universally better for every task.

7.2.2 Implications

From the results, we derived the following possible implications for (1) direct natural hand interaction in AR, (2) multimodal interaction in AR, and (3) natural interaction in AR.

7.2.2.1 Implication for Direct Natural Hand Interaction in AR

G-Shell demonstrated that direct natural hand interaction can offer a high level of usability, which was confirmed by the average usability rating scores for all tasks being above 5 points and subjective feedback of *Natural and intuitive to grasp*. The participants could directly manipulate virtual objects as if they were real, and expressed that it was believable, enjoyable and immersive.

It was observed that G-Shell was good for bimanual manipulation of single or a small group of objects. Nonetheless, direct manipulation of an object has limits with respect to the object's size and how cluttered the scene is. It was observed that some participants had difficulty manipulating objects that were too small, or had difficulty when the objects were cluttered together, as pointed out in *Smaller is harder*. One solution for this may be to provide zoom functionality where the whole scene can be resized and manipulation can be performed at a more manageable scale. The following design recommendations are provided:

Free those hands. Use direct natural hand interaction to improve naturalness, intuitiveness, and interactivity of the AR interface.

Zoom the world. Allow zooming to scale the scene so that the user's hands can manipulate objects easily.

7.2.2.2 Implication for Multimodal Interaction in AR

G-Speech provided interaction that combined gesture and speech input which offered a high level of usability, as shown by usability ratings where the overall rating was above 4.5 for relocation tasks and 6 for scaling, and user feedback of *easy and intuitive to resize* and *better control and so more precision*.

Indirect manipulation offers remote interaction where the hands do not needed to interact directly with the object, which can be beneficial when the interaction space is cluttered or small. It was found that G-Speech is effective at both single and multiple object manipulation, and there is no limit to the number of objects which can be manipulated at the same time. However, separating the interaction from the object has drawbacks, and it was observed that inexperienced participants were not able to predict the result of some indirect action as described in *difficult imagining an indirect rotation*. We suggest using a surrogate object close to the user but away from the scene, so that the user can interact directly with the surrogate to control the remote object.

Our design recommendations are:

Redundancy. Use several speech commands for the same action for better usability

Telekinesis. Use indirect manipulation to remotely interact with distant objects or many objects at the same time.

Surrogate. Provide a surrogate object to take the advantage of direct manipulation and remote interaction.

7.2.2.3 Implication for Natural Interaction for AR

Although the majority of participants liked the 6 *dof* movement offered by G-Shell, in certain case, more control was desired as mentioned by a participant in

better control and so more precision. Therefore it is a recommendation that the interface should offer choices for simplified actions as well as combined actions.

The results showed that G-Shell and G-Speech both had strengths and weaknesses. For natural interaction, it is challenging to design an interface suitable for every user and task, such as the example in *to speak or not to speak.* Therefore it is proposed that to enhance usability and user experience, offer complementary interactions within a single application so that the user has a choice. Our design recommendations are:

Divide and conquer. Provide both combined and divided actions such as translation, rotation, and both for better control.

Two is greater than one. Offer a choice of interaction techniques that complement each other.

7.2.3 Limitations

Although the study was designed to be as robust as possible, there were some limitations that are outlined in the following sections.

7.2.3.1 Limited Accuracy in Tracking Resolution

There are limits in tracking resolution and speech and gesture recognition accuracy in the 3Gear and Intel Perceptual Computing SDKs. We attempted to reduce the effect of these limitations as much as possible by calibrating the hand tracker for each user and providing commands that worked well for each subject. During the experiment, it was observed that typically only a small number of errors, below 5%, were encountered in a session regardless of interaction technique, reflecting the high usability rating for both interaction techniques.

7.2.3.2 Only Two Tasks were considered

Another limitation was that only two tasks, moving and scaling, were compared. However, as explained in Section 7.1.1.5, both of these tasks involved several fundamental actions (e.g. grasping, pointing, etc.) that are required in other, more complicated interaction.

7.3 Lessons Learnt

In this section we present the lessons we have learnt during this study.

7.3.1 Provide Multiple Modalities to Complement Each Other

In this study, we have demonstrated that both interaction techniques have their merits and weaknesses. For example, a direct natural hand interaction can offer a high level of interactivity in an area close to the user. This close range interaction literally offers a hands on manipulation that is familiar to users based on their interaction with physical objects in the real world. Conversely, an indirect manipulation technique can offer an interaction at a distance. Users can control objects without needing direct interaction by issuing verbal commands and performing contextual gestures. For example, gesture and speech can be used to move an object where the user is pointing as if ordering a virtual helper to perform the task. We believe it is beneficial to offer both methods of interaction in a single application. To build on the previous example, the user could get distant objects brought close using gesture and speech, and then directly manipulate the object when it's at close proximity.

Furthermore, different interaction techniques can complement and address the weakness of the other, especially in environments which don't conform to experimental conditions. For example, in a loud environment, it is unlikely that

the user can use a speech input to execute the task. Likewise, if the user's hands are occupied, it is unlikely that the user can easily use direct manipulation. Therefore user should be able to choose the interaction that work best for them given their current circumstances.

7.3.2 Multiple Choices of the Same Interaction and Commands

For better usability, choices for different actions and commands should be offered to execute the same interaction. As we have learnt, user preferences can vary greatly for the choices of actions or commands within the same interaction technique. To improve intuitiveness and guessability, multiple options within the same modality should be supported.

7.3.3 Options for Combined or Separated Actions

From the study, it was found that users find it useful to be able to operate with different level of control for different tasks, depending on user's priority of efficiency or accuracy. For example, to move an object from one location to another quickly, the user can use a 6 *dof* move to achieve both translation and rotation of the object at the same time. However, once the object is at its targeted location, the user may want to adjust only its orientation without affecting its position. By offering both combined and separated actions, users can decide and execute the appropriate action suited to their needs. This should improve user satisfaction, as well as efficiency and accuracy.

7.4 Conclusion

A comparison study between two natural interaction techniques was conducted for object relocation with both translation and rotation in 3D space, and uniform scaling of varied object shapes and size. The first technique, G-Shell, was based

on our proposed method of direct manipulation using natural hand interaction, while the second technique, G-Speech, was based on multimodal gesture-speech input. Using G-Shell, we demonstrated that a direct natural hand interaction technique can be natural, intuitive, interactive and precise. With G-Speech, we showed that ease of use and control is achievable for interactions without direct contact. G-Shell is better for object relocation while G-Speech was easier to use for uniform scaling.

From the findings, we recommend that both interaction techniques be combined in a single AR framework to improve usability and enhance user experience. Moreover, multiple options to perform the same interaction should also be provided for redundancy and help promote guessability. Integrated and divided actions should also be offered so that users can choose the level of control that suits their needs.

In the next chapter, we report the progress on our research goals. We summarize the high level design guidelines from the lessons learnt throughout this research. Finally, we give a discussion on natural hand interaction for AR and possible future work.

Part IV
Discussion, Further Work and Conclusions

Chapter 8
Discussion and Future Work

In Chapter 1 of this thesis, we introduced the topic of this thesis and proposed a number of goals that we hoped to achieve. The work presented in the following chapters focused on exploration and understanding of natural hand interaction in AR (Chapters 3 and 4), and development, implementation and evaluation of a multimodal AR framework and direct and indirect natural interaction techniques (Chapters 5, 6 and 7).

In this chapter, we revisit our original goals to determine if we were successful in Section 8.1. In Section 8.2 we present a summary of design guidelines for natural interaction in AR that can be derived from this work. A general discussion of the current state of natural hand interaction for AR is given in Section 8.3 and we explore some future research directions in Section 8.4.

8.1 Progress on Research Goals

As discussed in the introduction of this thesis, our two primary goals were *to understand* and then *enhance natural hand interaction in AR*. These primary goals can be decomposed into six subgoals as follows:

1. Understand the best practices and limitations of the technology in current AR interfaces and interaction techniques.

2. Observe user interaction with an AR system that offers environment awareness and physically-based interaction, and use this information to determine the characteristics of affordance of such an AR interface.

3. Learn hand gestures that are preferable and easy to perform in AR from users and create design guidelines from the findings.

4. Develop a gesture interface that utilizes depth sensing technology for hand tracking and recognition.

5. As a baseline comparison for the new interaction technique, develop an AR framework that supports natural hand and gesture-speech interaction as the primary inputs.

6. Evaluate and compare the proposed natural hand interaction technique to the multimodal interaction technique.

The first subgoal was achieved by conducting a literature review. Chapter 2 covered existing research in AR and interaction techniques both before and after the widespread adoption of consumer depth sensors.

The second subgoal was satisfied by making observations and collecting feedback from demonstrating two AR systems as described in Chapter 3. These systems support basic natural hand interaction in an environmental aware and physics simulated environment. The two systems were demonstrated in two different configurations; (1) face-to-face collaborative tabletop, (2) mobile tablet.

The third subgoal was completed by conducting a guessability study on hand gestures in AR in Chapter 4, which yielded the first user-defined gesture set for AR. Characterization of the preferred hand gestures lead to a deeper understanding of the user design process and the implications of the study for AR, gesture interface, and gesture recognition were discussed.

The fourth subgoal was accomplished by the implementation of the gesture interface and the discussion of the lessons learnt during the development in

Chapter 5. Lessons learnt from the development of this interface contributed to the development of our AR framework. We have learnt that hand tracking and classification is crucial to the success of the natural hand interaction.

The fifth subgoal was fulfilled with the design and demonstration of an AR system implemented using a novel AR framework, G-SIAR. The design and implementation of both hardware and software components of G-SIAR were summarized in Chapter 6.

The final subgoal was realized with the design and evaluation of G-Shell, a novel natural hand interaction technique, and G-Speech, a multimodal interaction technique, in Chapters 6 and 7. Both G-Shell and G-Speech were rated highly in terms of usability. Comparing G-Shell to G-Speech yielded both quantitative findings in performance and qualitative findings in user preference. The implications of this for direct natural hand interaction, multimodal interaction, and the overall natural interaction in AR are discussed in Chapter 7.

Through the achievement of all the subgoals, our primary goals of *understanding natural hand interaction* (covered by subgoals 1, 2, and 3) and *enhancing natural hand interaction* (covered by subgoals 4, 5, and 6) have been satisfied.

8.2 Summary of Design Guidelines

8.2.1 Physical is Natural in AR

Physical actions are inherent to interaction in AR. The main goal of AR is to display virtual content as if it is part of the real world, and it is important that the interaction techniques for AR interfaces support this goal. For this reason, tangible interfaces are a natural choice for interaction in AR. In Chapter 3 we presented two novel AR systems, and found that offering environment awareness and physics-enabled simulation in a system can provide a natural and

intuitive method of interaction, and increase the realism and immersiveness of the experience.

Physical hand manipulation is natural in AR. Interaction with physical objects in the real world is primarily conducted with physical manipulation. In Chapter 4 we conducted a guessability study to determine the preferred methods of interaction with virtual objects and found that, for tasks which affect the physical aspects of an objects such as moving, scaling, etc, users again prefer physical gestures for manipulation. It is important to support this type of interaction in a system to allow users to intuitively interaction with virtual content.

8.2.2 Beyond Physical Escalates Complexity

Complex constructs lead to complex actions. Another finding from our guessability study in Chapter 4 was that it was difficult for users to describe abstract concepts such as copy, cut, and paste, using physical gestures. For these complex constructs metaphorical or symbolic gestures were often used. The gestures used can vary considerably from one user to another depending on personal preference, and even social and cultural background.

As there are many possible metaphors that can be used for a single concept, agreement on just one metaphoric gesture can be difficult. Therefore if metaphorical or symbolic gestures are to be used, it should be determined if there is a universally accepted gesture that could be used, or otherwise whether custom gesture sets can be used for different groups of users. Multiple gestures could also be mapped to the same function, which promotes guessability and intuitiveness.

Complexity reduction by using the affordance of real world objects. The affordances of objects in the real world are well known to us, and this knowledge can be used to intuitively communicate affordances in the virtual world. For example, doorknobs are designed for opening and shutting door, and are operated by twist and push or pull. By visually mimicking the appearance of a doorknob, users will perform this familiar interaction intuitively. This behavior could be seen in our study in Chapter 4, where many users imagined a virtual dial to control simulation speed with a twisting motion in one direction for play/increase speed, and the opposite direction for stop/decrease speed.

8.2.3 Auxiliary Enhancement are Crucial

Aggregated and segregated commands are important to different aspects of performance and should both be offered. The precision and accuracy of a system depends on its sensing components, and with the current depth sensing technology noise is common place and resolution is still limited, which can affect interaction satisfaction.

Regardless of how crucial these factors are to the application, the design of the interaction should not create frustration and offer alternative methods of solving problems. Complex tasks can be divided when necessary, for instance in our evaluation study in Chapter 7, users stated that they would like if it was possible to separate translation and rotation actions to allow for more control, but still use the combined action to increase efficiency when accuracy is not a necessity. One possible solution would be to allow the user to translate and rotate an object at the same time, but once it is close to the target location the user can translate and rotate separately without having to issue an explicit command.

Minor attention to details can make a major difference in better user experience. Direct interaction with virtual objects can be difficult due to the absence of haptic feedback. If haptic feedback cannot be offered, visual and audio feedback are crucial to the improvement of user experience. Visual occlusion of the user's hand and virtual objects is crucial for the user's depth perception and directly impacts the interaction. Once the hand makes a contact with the object, feedback should be given so the user knows they are touching the object. Furthermore, different modes or actions that the user is performing should be clearly indicated in an unobtrusive manner, for example, different colored outlines on the object being manipulated to indicate translation or rotation. In Chapter 6 we present our multimodal framework G-SIAR, which is built to support these feedback mechanisms to maximize user satisfaction.

8.2.4 The More the Merrier

Multimodality is a useful feature for AR interface. Our findings from the experiment described in Chapter 7 showed that different users have different preferences and, in terms of natural interaction, what is natural for one user might not be for another. This aligns with the guidelines covered in Section 8.2.3, that suggests offering both aggregate and segregate commands to give users the freedom to choose the appropriate method to carry out their task in different situations. Offering multimodality means that users have more options to choose what is the most natural for them and an appropriate modality to use in the current environment.

8.3 Discussion of Natural Hand Interaction for Augmented Reality

In Chapter 4, we conducted a guessability study on user-defined gestures to learn how users would interact using natural hand interaction in AR without the

limitations of the current hand tracking technology. In the experimental setup, the participants sat in front of a table where the interaction space was located. The interaction space was within arm's reach and therefore the gestures elicited could be performed directly on the targeted objects. A resulting limitation of this setup is that the results may not be transferable to interaction spaces larger than arms reach and with objects are greater distances apart. However, we feel that this limitation does not invalidate our findings as the physical gestures that we elicited can still be applied to larger interaction spaces by allowing the user to move closer to the object they wish to interact with or bringing the object closer to the user. Still, it is valuable to explore what other methods the participants may imagine these situations.

The elicited gesture set was not meant to be the one and only set for all future applications, but was designed to serve as a guideline for designing natural hand interaction. We believe that for every new set of participants who carry out the study, there will be new gestures with high agreement scores. This also does not invalidate our findings, and we recommend in our design guidelines that multiple choices of the same action should be offered to improve intuitiveness and guessability. Ideally, the gestures database should support all possible gestures but to increase efficiency and accuracy by reducing search space, users should have the freedom to choose their favorite gestures for interaction.

The motivation behind the study to compare natural hand interaction and gesture-speech interaction in terms of direct and indirect manipulation, described in Chapter 7, originated from the original debate between Shneiderman and Maes (1997) over direct manipulation and interface agents, as discussed in Section 2.1.2.1. It was found that user interface and interaction design are strongly coupled and care must be taken in choosing an appropriate method of interaction that is supported by the interface for better usability and performance. We found from our study that each interaction technique had its

merits and weaknesses and the two techniques complemented each other well. These findings are in agreement with a compromised approach called the Mixed Initiative (MI) that was proposed by Horvitz (1999), where both techniques coexist simultaneously. We do not believe in a one-size-fits-all solution but we think that user should have the freedom to decide whether to take the complete control or delegate tasks to the machine.

8.4 Future Work

In this section, we present future work aimed to address limitations in this research (Section 8.4.1), as well as possible research direction beyond the scope of this thesis (Section 8.4.2).

8.4.1 Addressing Limitations in this Work

In Chapter 4 we presented a guessability study that aimed to get user feedback on gesture preferences. This study had two main limitations, as discussed in Section 4.4.4. The first limitation was that the experiment was conducted on a tabletop space, and some of our findings may not be directly applicable to a larger interaction area such as a full room. We propose conducting another study to elicit gestures in different settings which may be used for AR. In addition, in the guessability study, we chose not to explore any social or cultural aspects of the elicited gestures. By considering this new dimension, deeper understanding into user preference and insights for designing gestures could be gained.

The gesture interface built and discussed in Chapter 5 was motivated by a lack of open source gesture interfaces available at the time of writing. Unfortunately difficulties with the hand tracking and classification components meant that the end result was not able to perform as well as commercial offerings, and so the 3Gear Nimble SDK was used for the user evaluation. By

integrating the latest research into our interface, it is possible it could equal or surpass the best commercial offerings.

Our AR Framework G-SIAR, presented in Chapter 6, used an in-place depth sensor for hand tracking and an image-based marker to calculate the viewing camera transformation. We would like to improve our framework so that it can offer mobility by mounting the depth sensor onto the HMD and using environmental tracking.

Following the usability study presented in Chapter 7, which compared natural hand interaction and gesture-speech interaction, we proposed that AR interface should offer multiple modalities to complement each other. We would like to validate this proposal by comparing the usability and performance of individual modality to combined modalities in a number of real world AR applications. In addition, as mentioned in Section 7.2.3.2, only two tasks, moving and scaling, were supported and examined in the study. More tasks should be compared using our interaction techniques to learn the strengths and weaknesses of each modality for these new tasks.

8.4.2 Beyond the Scope of this Thesis

In addition to physical manipulation and gestures mapped to commands, Morgado (2014) proposed the Shamanic interface where the system can understand beyond mimicry and non-kinesthetic gestures that support gestures that follow social and cultural influence. This vision might not be easily achievable in the near future but it is an interesting concept that researchers and developers can aim to achieve.

One limitation when natural hand interaction is used for direct manipulation of the physical objects that is the lack of haptic or tactile feedback. It would be ideal if these feedback mechanisms can be provided with minimal inconvenience to the user. A number of recent works have been exploring this

area, delivering touch sensation in mid-air utilizing sharp bursts of air (Sodhi et al., 2013) as well as ultrasound (Long et al., 2014).

Beyond in the air feedback, AR interfaces can utilize ordinary objects to provide a physical proxy that makes use of tactile sensation. A combination of external sensing, for example vision-based tracking using depth cameras, and internal sensing, for example accelerometers and gyroscopes, can turn any physical object into a high fidelity input device.

Chapter 9
Conclusion

Natural hand interaction provides an intuitive way for users to interact in both real and virtual environments at the same time, which makes it ideal for interaction in Augmented Reality. It was this potential that motivated this research in how to both better understand and enhance natural hand interaction in AR. To improve understanding, we conducted an in-depth review of past research, found the characteristics of affordance, and determined user preference using a guessability study. To enhance natural hand interaction, we implemented a gesture interface, integrated software components that we developed into a new AR framework, and introduced a novel natural hand interaction technique and a multimodal gesture and speech interaction technique.

After reviewing a wide range of research areas in AR including software architecture, applications, evaluation methods, and collaboration, we compiled a comprehensive review of interaction techniques in AR before and after the prevalence of consumer depth sensor. We found a number of shortcomings and technical limitations in previous research, and determined that further research was required to better understand user experience and preference for natural hand interaction in AR.

Two AR systems were developed for collaborative tabletop and mobile gaming settings to explore the characteristics of affordance of natural hand interaction. These systems support environment awareness and physically-based interaction utilizing a depth sensor. From analyzing observation and feedback from public demonstrations, it was found that direct physical

interaction using hands and physical objects was natural and intuitive, leading to the conclusion that physical actions are inherent to interaction in AR.

Although prior research had demonstrated hand gestures in AR, there was no consensus on how this interaction technique can best serve users. To learn user preference for natural hand interaction for AR, a guessability study was conducted. The participants wore a HMD and were shown animations of 40 tasks in AR. The participants provided the hand gesture they felt was most suitable and easy to perform for each task. From the 800 collected gestures, the 44 gestures with highest agreement scores were chosen as the user-defined gesture set for AR. This study yielded the first user-defined gesture set in AR, a taxonomy of gestures in AR, and the design recommendations and implications.

From the guessability study, it was found that a wide range of expressive hand gestures were elicited but could not be supported due to limitations in the underlying technologies of hand recognition and tracking. To address this issue, we developed a gesture interface to support these hand gestures which comprises of five major components including: the hardware interface that utilizes depth sensors, the segmentation and tracking component that detects and segments hands, the classification component that uses a random forest algorithm for partitioning the hand into sub-regions, the modeling component for physical simulation, and the gesture recognition component for interpreting gestures.

A number of challenges were faced during the development of the gesture interface. It is difficult to determine the hand's initial position. The synthetic hand data set should be limited to a small subset that focuses on the hand pose to be recognized. Determination of the training parameters is a trial and error process that can be sped up by parallel computing. The random forest classification technique is sensitive to rotation and scaling of the input image. Most challenging though was getting accurate results from the hand tracking

and classification process. Due to these limitations, for our final AR framework we opted to use a commercial hand tracking and classification solution to improve the user experience.

We proposed G-SIAR, a multimodal augmented reality framework, which supports natural interaction as the primary input. The framework provides an immersive and large interaction space, physical simulation, real-time and seamless object creation and interaction, and realistic rendering with shadows and hand occlusion. Within the framework, two interaction techniques were implemented, Grasp-Shell (G-Shell), a natural hand interaction technique and Gesture-Speech (G-Speech), a multimodal gesture-speech interaction technique.

In our final study, we evaluated and compared the usability of two natural interaction techniques implemented in our framework, G-Shell and G-Speech. The study contained three tasks: single object relocation, multiple object relocation, and uniform scaling. It was found that G-Shell was better for object relocation tasks while G-Speech was better for uniform scaling. This supported our hypotheses that there was a difference in performance, usability, and preference between these two interaction techniques and confirmed our beliefs that each technique had its strengths and weaknesses. We recommend that future AR interfaces provide multiple modalities to complement each other so that users have the freedom to choose the interaction that best suits the circumstance.

This thesis makes six major contributions to the field of natural interaction in AR, they are as follows:

1. A literature review of augmented reality and interaction techniques. The focus of this review is on the history of AR and research in interaction techniques for AR before and after the wide adoption depth sensing.

2. Two AR systems supporting collaborative tabletop and mobile settings. Both systems featured environmental awareness, physically-based interaction, and basic natural hand interaction. We summarized the lessons learnt from the design and development process including observations and feedback from public demonstrations.

3. The findings from a guessability study on user-defined gestures for AR, which yielded a number of minor contributions including: the first comprehensive set of user-defined gestures for AR, classification of elicited gestures based on a gesture taxonomy for AR, the agreement scores of gestures for selected tasks and their subjective rating, the qualitative findings from the design process, and the implications of this work for AR, gesture interfaces, and gesture recognition.

4. A gesture interface comprised of five major components including: the hardware interface, the segmentation and tracking component, the classification using random forests component, the physical simulation modeling component, and the gesture recognition component. The lessons learnt from the development were shared.

5. The G-SIAR multimodal augmented reality framework that offers natural interaction as a native input. The framework supports highly immersive and encompassing viewing coverage of the interaction space, physics-enabled simulation, real-time and seamless object creation and interaction, and realistic rendering with shadows and hand occlusion.

6. Two interaction techniques: Grasp-Shell, a direct natural hand interaction technique and Gesture-Speech, an indirect multimodal gesture-speech interaction technique. A usability study of these two

interaction techniques for three tasks including single objection relocation, multiple object relocation, and uniform resizing, gave insights into performance, usability, and preference as well as the strengths and weaknesses of each technique.

With these contributions, future research can directly benefit from the lessons that we shared, the limitations of current research that we summarized, the implementation of our AR framework and interaction techniques, the design guidelines that we have given, and the direction for future work that we discussed.

References

Akman, O., Poelman, R., Caarls, W., & Jonker, P. (2013). Multi-cue hand detection and tracking for a head-mounted augmented reality system. *Machine Vision and Applications, 24*(5), pp 931-946.

Aleotti, J., Denaro, F., & Caselli, S. (2010). Object manipulation in visuo-haptic augmented reality with physics-based animation. *The 19th IEEE International Symposium on Robot and Human Interactive Communication (ROMAN'10)*, pp 13-15.

ARToolKit. (2015). Retrieved from www.hitl.washington.edu/artoolkit.

Azuma, R. (1997). A Survey of Augmented Reality. *Presence, 6*, pp 355-385.

Azuma, R., Baillot, Y., Behringer, R., Feiner, S., Julier, S., & MacIntyre, B. (2001). Recent advances in augmented reality. *IEEE Computer Graphics and Applications, 21*, pp 34-47.

Barakonyi, I., & Schmalstieg, D. (2007). Ubiquitous animated agents for augmented reality. *The 5th IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*, pp 145-154.

Barakonyi, I., & Schmalstieg, D. (2008). Augmented reality agents for user interface adaptation. *Computer Animation and Virtual Worlds, 19*, pp 23-35.

Batlle, J., Mouaddib, E., & Salvi, J. (1998). Recent progress in coded structured light as a technique to solve the correspondence problem: a survey. *Pattern Recognition, 31*(7), 963-982.

Bay, H., Tuytelaars, T., & Van Gool, L. (2006). SURF: Speeded up robust features. *The 9th European Conference on Computer Vision (ECCV'09)*, pp 404-417.

Benko, H., Ishak, E. W., & Feiner, S. (2004). Collaborative mixed reality visualization of an archaeological excavation. *Proceedings of the Third IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'04)*, pp 132-140.

Benko, H., Jota, R., & Wilson, A. (2012). MirageTable: freehand interaction on a projected augmented reality tabletop. *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems (CHI'12)*, pp 199-208.

Berger, M. O. (1997). Resolving occlusion in augmented reality: a contour based approach without 3D reconstruction. *Proceedings of IEEE Computer Society on Computer Vision and Pattern Recognition*, pp 91-96.

Billinghurst, M. (2008). Usability testing of augmented / mixed reality systems. *ACM SIGGRAPH ASIA 2008 Courses*.

Billinghurst, M., Grasset, R., & Looser, J. (2005). Designing augmented reality interfaces. *Computer Graphics (ACM), 39*, pp 17-22.

Billinghurst, M., Grasset, R., & Seichter, H. (2009). Tangible Interfaces for Ambient Augmented Reality Applications. pp 387-396.

Billinghurst, M., & Kato, H. (2002). Collaborative augmented reality. *Communications of the ACM, 45*, pp 64-70.

Billinghurst, M., Kato, H., & Myojin, S. (2009). Advanced interaction techniques for augmented reality applications. *3rd International Conference on Virtual and Mixed Reality (VMR'09)*. pp 13-22.

Billinghurst, M., Kato, H., & Poupyrev, I. (2001a). Collaboration with tangible augmented reality interfaces. *Proceedings of Human Computer Interaction International (HCII'01)*.pp 797-803.

Billinghurst, M., Kato, H., & Poupyrev, I. (2001b). The MagicBook - moving seamlessly between reality and virtuality. *Computer Graphics and Applications, IEEE, 21*(3), 6-8.

Billinghurst, M., Kato, H., & Poupyrev, I. (2008). Tangible augmented reality. *ACM SIGGRAPH ASIA 2008 courses.*

Billinghurst, M., Poupyrev, I., Kato, H., & May, R. (2000). Mixing realities in Shared Space: an augmented reality interface for collaborative computing. *Proceedings of International Conference on Multimedia and Expo*, pp 1641-1644.

Billinghurst, M., Weghorst, S., & Furness, T. (1998). Shared space: An augmented reality approach for computer supported collaborative work. *Virtual Reality, 3*(1), 25-36.

Bimber, O., & Raskar, R. (2005). *Spatial Augmented Reality: Merging Real and Virtual Worlds*: A. K. Peters, Ltd.

Bolt, R. A. (1980). Put-that-there: Voice and gesture at the graphics interface. *SIGGRAPH Comput. Graph., 14*(3), 262-270.

Bonanni, L., Lee, C.-H., & Selker, T. (2005). A framework for designing intelligent task-oriented augmented reality user interfaces. *Proceedings of the 10th international conference on Intelligent user interfaces*, pp 317-319.

Bradski, G. R., & Kaehler, A. (2008). *Learning OpenCV*. Farnham; Cambridge: O'Reilly.

Breen, D. E., Rose, E., & Whitaker, R. T. (1995). Interactive Occlusion and Collision of Real and Virtual Objects in Augmented Reality. *Technical Report ECRC-95-02, ECRC, Munich, Germany*.

Breiman, L. (2001). Random Forests. *Mach. Learn., 45*(1), 5-32.

Broll, W., Lindt, I., Ohlenburg, J., Wittkamper, M., Yuan, C., Novotny, T., . . . Strothmann, A. (2004). ARTHUR: A Collaborative Augmented Environment for Architectural Design and Urban Planning. *Journal of Virtual Reality and Broadcasting, 1*.

Bruhn, A., Weickert, J., Feddern, C., Kohlberger, T., & Schnorr, C. (2005). Variational optical flow computation in real time. *IEEE Transactions on Image Processing, 14*(5), pp 608-615.

Buchanan, P., Seichter, H., Billinghurst, M., & Grasset, R. (2008). Augmented reality and rigid body simulation for edutainment: The interesting mechanism - An AR puzzle to teach Newton physics. *International Conference on Advances in Computer Entertainment Technology (ACE'08)*, pp 17-20.

Buchmann, V., Violich, S., Billinghurst, M., & Cockburn, A. (2004). FingARtips - Gesture based direct manipulation in augmented reality. *Proceedings of 2nd International Conference on Computer Graphics and Interactive Techniques (GRAPHITE'04)*, pp 212-221.

Bullet Physics Engine. (2015). Retrieved from http://www.bulletphysics.org.

Butz, A., Hollerer, T., Feiner, S., MacIntyre, B., & Beshers, C. (1999). Enveloping users and computers in a collaborative 3D augmented reality. *Proceedings of 2nd IEEE and ACM International Workshop on Augmented Reality (IWAR '99)*, pp 35-44.

Clark, A. J., Green, R. D., & Grant, R. N. (2008). Perspective correction for improved visual registration using natural features. *Proceedings of 23rd International Conference Image and Vision Computing New Zealand (IVCNZ'08)*, pp 1-6.

Coutaz, J., Nigay, L., Salber, D., Blandford, A., May, J., & Young, R. M. (1995). Four easy pieces for assessing the usability of multimodal interaction: the CARE properties. *Proceedings of 5th International Conference on Human-Computer Interaction (INTERACT'95)*, pp 115-120.

Datcu, D., Cidota, M., Lukosch, H., & Lukosch, S. (2014). On the usability of augmented reality for information exchange in teams from the security domain. *Proceedings of IEEE Joint Intelligence and Security Informatics Conference (JISIC'14)*, pp 160-167.

Datcu, D., & Lukosch, S. (2013). Free-hands interaction in augmented reality. *Proceedings of the 1st symposium on Spatial user interaction (SUI'13)*, pp 33-40.

Datcu, D., Lukosch, S., & Brazier, F. (2015). On the Usability and Effectiveness of Different Interaction Types in Augmented Reality. *International Journal of Human-Computer Interaction, 31*(3), 193-209.

Dell Venue 8 7000 Series. Retrieved from http://www.dell.com/us/p/dell-venue-8-7840-tablet/pd.

Dumas, B., Lalanne, D., & Oviatt, S. (2009). Multimodal Interfaces: A Survey of Principles, Models and Frameworks *Human Machine Interaction - Research Results of the MMI Program*, pp. 3-26.

Dünser, A., Grasset, R., & Billinghurst, M. (2008). A survey of evaluation techniques used in augmented reality studies. *ACM SIGGRAPH ASIA 2008 courses*.

Engelbart, D. C., & English, W. K. (1968). A research center for augmenting human intellect. *Proceedings of fall joint computer conference, part I*, pp 395-410.

Erol, A., Bebis, G., Nicolescu, M., Boyle, R. D., & Twombly, X. (2007). Vision-based hand pose estimation: A review. *Computer Vision and Image Understanding, 108*(1-2), 52-73.

Falahati, S. (2013). *OpenNI Cookbook*: Packt Publishing, Limited.

Feiner, S., Macintyre, B., & Seligmann, D. (1993). Knowledge-based augmented reality. *Commun. ACM, 36*(7), 53-62.

Fernandes, B., & Fernandez, J. (2009). Bare hand interaction in tabletop augmented reality. *SIGGRAPH '09: Posters*.

Fischer, J., Huhle, B., & Schilling, A. (2007). Using Time-of-Flight Range Data for Occlusion Handling in Augmented Reality. *Eurographics Symposium on Virtual Environments (EGVE)*, 109-116.

Guangqi, Y., Corso, J. J., Hager, G. D., & Okamura, A. M. (2003). *VisHap: augmented reality combining haptics and vision. Proceedings of IEEE International Conference on Systems, Man and Cybernetics (SMC'03)*, pp 3425-3431.

HandVu. (2015). Retrieved from http://www.movesinstitute.org/~kolsch/HandVu/HandVu.html.

Harders, M., Bianchi, G., & Knoerlein, B. (2007). Multimodal augmented reality in medicine. *Proceedings of the 4th International Conference on Universal Access in Human-Computer Interaction (UAHCI'07)*, pp 652-658.

Harrison, C., Benko, H., & Wilson, A. D. (2011). OmniTouch: wearable multitouch interaction everywhere. *Proceedings of the 24th annual ACM symposium on User interface software and technology (UIST'11)*, pp 441-450.

Havok Physics. (2015). Retrieved from http://www.havok.com.

Heidemann, G., Bax, I., & Bekel, H. (2004). Multimodal interaction in an augmented reality scenario. *Proceedings of the Sixth International Conference on Multimodal Interfaces (ICMI'04)*, pp 53-60.

Hilliges, O., Izadi, S., Wilson, A. D., Hodges, S., Garcia-Mendoza, A., & Butz, A. (2009). Interactions in the air: adding further depth to interactive tabletops. *Proceedings of the 22nd annual ACM symposium on User interface software and technology (UIST'09)*, pp 139-148.

Hilliges, O., Kim, D., Izadi, S., Weiss, M., & Wilson, A. (2012). HoloDesk: direct 3d interactions with a situated see-through display. *Proceedings of the ACM annual conference on Human Factors in Computing Systems (CHI'12)*, pp 2421-2430.

Hornecker, E., & Dünser, A. (2009). Of Pages and Paddles: Children's Expectations and Mistaken Interactions with Physical-Digital Too. *Interacting with Computers*(1-2), 95-107.

Horvitz, E. (1999). Principles of mixed-initiative user interfaces. *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*, pp 159-166.

Hurst, W., & Van Wezel, C. (2011). Multimodal interaction concepts for mobile augmented reality applications. *Proceedings of the 17th Multimedia Modeling Conference (MMM'11)*, pp 157-167.

Intel RealSense. (2015). Retrieved from http://www.intel.com/content/www/us/en/architecture-and-technology/realsense-overview.html

Irawati, S., Green, S., Billinghurst, M., Duenser, A., & Ko, H. (2006). An evaluation of an augmented reality multimodal interface using speech and paddle gestures. *Proceedings of the 16th International Conference on Artificial Reality and Telexistence (ICAT'06)*, pp 272-283.

Irawati, S., Green, S., Billinghurst, M., Duenser, A., & Ko, H. (2007). Move the couch where? : Developing an augmented reality multimodal interface. *Proceedings of the 5th IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'05)*, pp 183-186.

Ishii, H. (2008). Tangible bits: Beyond pixels. *Proceedings of the 2nd International Conference on Tangible and Embedded Interaction, (TEI'08)*, pp xv-xxv.

Ishii, H., & Ullmer, B. (1997). Tangible bits: towards seamless interfaces between people, bits and atoms. *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp 234-241.

Ishii, H., Underkoffler, J., Chak, D., Piper, B., Ben-Joseph, E., Yeung, L., & Kanji, Z. (2002). Augmented urban planning workbench: overlaying drawings, physical models and digital simulation. *Proceedings of the IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'02)*, pp 203-211.

Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., & Fitzgibbon, A. (2011). KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera. *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pp 559-568.

Jintae, L., & Kunii, T. L. (1993). Constraint-based hand animation. *Proceedings of Computer Animation '93*, pp 110-127.

Jones, B., Sodhi, R., Murdock, M., Mehra, R., Benko, H., Wilson, A., & Shapira, L. (2014). RoomAlive: magical experiences enabled by scalable, adaptive projector-camera units. *Proceedings of the 27th annual ACM symposium on User interface software and technology (UIST'14)*, pp 637-644.

Jones, B. R., Benko, H., Ofek, E., & Wilson, A. D. (2013). IllumiRoom: peripheral projected illusions for interactive experiences. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'13)*, pp 869-878.

Julier, S., Livingston, M. A., Swan, J. E., Baillot, Y., & Brown, D. (2003). *Adaptive User Interfaces in Augmented Reality*.

Kaiser, E., Olwal, A., McGee, D., Benko, H., Corradini, A., Li, X., & Feiner, S. (2003). Mutual disambiguation of 3D multimodal interaction in augmented and virtual reality. Proceedings of the Fifth International Conference on Multimodal Interfaces (ICMI'03), pp 12-19.

Kato, H., Billinghurst, M., Poupyrev, I., Imamoto, K., & Tachibana, K. (2000). Virtual object manipulation on a table-top AR environment. *Proceedings IEEE and ACM International Symposium on Augmented Reality (ISAR 2000)*, pp 111-119.

Keskin, C., Kirac, F., Kara, Y. E., & Akarun, L. (2011). Real time hand pose estimation using depth sensors. *Proceedings of the IEEE International Conference on Computer Vision Workshops, (ICCV'11 Workshops)*, pp 1228-1234.

Kim, D., Hilliges, O., Izadi, S., Butler, A. D., Chen, J., Oikonomidis, I., & Olivier, P. (2012). Digits: freehand 3D interactions anywhere using a wrist-worn gloveless sensor. *Proceedings of the 25th annual ACM symposium on User interface software and technology*, pp 167-176.

Kiyokawa, K., Iwasa, H., Takemura, H., & Yokoya, N. (1998). Collaborative immersive workspace through a shared augmented environment. Proceedings of the Intelligent Systems in Design and Manufacturing, pp 2-13.

Kiyokawa, K., Takemura, H., & Yokoya, N. (1999, 1999). A collaboration support technique by integrating a shared virtual reality and a shared augmented reality. *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC'99)*, pp 48-53.

Kolsch, M., Bane, R., Hollerer, T., & Turk, M. (2006). Multimodal interaction with a wearable augmented reality system. *IEEE Computer Graphics and Applications, 26*, 62-71.

Kolsch, M., & Turk, M. (2004, 27-02 June 2004). Fast 2D Hand Tracking with Flocks of Features and Multi-Cue Integration. *Proceedings of the Conference on Computer Vision and Pattern Recognition Workshop (CVPRW '04)*, pp 158.

Lalanne, D., Nigay, L., Palanque, P., Robinson, P., Vanderdonckt, J., & Ladry, J.-F. (2009). Fusion engines for multimodal input: A survey. *Proceedings of the International Conference on Multimodal Interfaces and the Workshop on Machine Learning for Multimodal Interfaces*, (ICMI-MLMI'09), pp 153-160.

LeapVR. (2015) Retrieved from https://www.leapmotion.com/product/vr.

Lee, G. A., Billinghurst, M., & Kim, G. J. (2004). Occlusion based interaction methods for tangible augmented reality environments. *Proceedings of the International Conference on Virtual Reality Continuum and its Applications in Industry* (VRCAI'04), pp 419-426.

Lee, H., Billinghurst, M., & Woo, W. (2011). Two-handed tangible interaction techniques for composing augmented blocks. *Virtual Reality, 15*(2), pp 133-146.

Lee, J., Olwal, A., Ishii, H., & Boulanger, C. (2013). SpaceTop: integrating 2D and spatial 3D interactions in a see-through desktop environment. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp 189-192.

Lee, J. Y., Rhee, G. W., & Seo, D. W. (2010). Hand gesture-based tangible interactions for manipulating virtual objects in a mixed reality environment. *International Journal of Advanced Manufacturing Technology, 51*, 1069-1082.

Lee, M. (2010). *Multimodal Speech-Gesture Interaction with 3D Objects in Augmented Reality Environments.* (Doctor of Philosophy), University of Canterbury, Christchurch, New Zealand.

Lee, M., & Billinghurst, M. (2008). A Wizard of Oz study for an AR multimodal interface. *Proceedings of the 10th international conference on Multimodal interfaces (ICMI'08)*, pp 249-256.

Lee, M., Billinghurst, M., Baek, W., Green, R., & Woo, W. (2013). A usability study of multimodal input in an augmented reality environment. *Virtual Reality, 17*(4), 293-305.

Lee, M., Green, R., & Billinghurst, M. (2008). 3D natural hand interaction for AR applications. *Proceedings of the 23rd International Conference Image and Vision Computing New Zealand, (IVCNZ'08).*

Lee, T., & Hollerer, T. (2007). Handy AR: Markerless inspection of augmented reality objects using fingertip tracking. *Proceedings of the 11th IEEE International Symposium on Wearable Computers* (ISWC'07), pp 83-90.

Lee, T., & Hollerer, T. (2009). Multithreaded Hybrid Feature Tracking for Markerless Augmented Reality. *Visualization and Computer Graphics, IEEE Transactions on, 15*(3), 355-368.

Lepetit, V., & Berger, M. O. (2000). Handling occlusion in augmented reality systems: a semi-automatic method. *Proceedings IEEE and ACM International Symposium on Augmented Reality (ISAR'00)*, pp 137-146.

Long, B., Seah, S.A., Carter, T., and Subramanian, S. (2014). Rendering volumetric haptic shapes in mid-air using ultrasound, *ACM Trans. Graph., 2014, 33, (6)*, pp. 1-10.

Looser, J., Grasset, R., & Billinghurst, M. (2007). A 3D Flexible and Tangible Magic Lens in Augmented Reality. *IEEE Computer Society*, pp 1-4.

MacIntyre, B., Gandy, M., Dow, S., & Bolter, J. D. (2005). DART: A toolkit for rapid design exploration of augmented reality experiences. *ACM Transactions on Graphics*, pp 932.

MacNamee, B., Beaney, D., & Qingqing, D. (2010). Motion in Augmented Reality Games: An Engine for Creating Plausible Physical Interactions in Augmented Reality Games. *International Journal of Computer Games Technology*.

Maes, P., & Kozierok, R. (1993). Learning interface agents. *Proceedings of the 11th National Conference on Artificial Intelligence*, pp 459-465.

Matsuda, K. (2010). *Domesti/City: The dislocated home in augmented space.* (M.Arch.), University College London, London. Retrieved from http://www.keiichimatsuda.com/thesis.php.

Maybury, M., & Wahlster, W. (1998). *Readings in Intelligent User Interfaces*. In M. Maybury & W. Wahlster (Eds.): Morgan Kaufmann Publishers.

Meta. Retrieved from https://www.getameta.com.

Microsoft Hololens. (2015). Retrieved from http://www.microsoft.com/microsoft-hololens.

Microsoft Kinect SDK. (2015) Retrieved from http://www.microsoft.com/en-us/kinectforwindows/.

Milgram, P., & Kishino, F. (1994). Taxonomy of mixed reality visual displays. *IEICE Transactions on Information and Systems, E77-D*, pp 1321-1329.

Mitra, S., & Acharya, T. (2007). Gesture Recognition: A Survey. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, 37*(3), pp 311-324.

Morgado, L. (2014). Cultural Awareness and Personal Customization of Gestural Commands Using a Shamanic Interface. *Procedia Computer Science, 27,* pp 449-459.

Newcombe, R. A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A. J., & Fitzgibbon, A. (2011). KinectFusion: Real-time dense surface mapping and tracking. *Proceedings of the 10th IEEE International Symposium on Mixed and Augmented Reality, (ISMAR'11),* pp 127-136.

Newton Dynamics. (2015). Retrieved from http://www.newtondynamics.com.

Nickolls, J., Buck, I., Garland, M., & Skadron, K. (2008). Scalable Parallel Programming with CUDA. *Queue, 6*(2), pp 40-53.

Nigay, L., & Coutaz, J. (1993). Design space for multimodal systems. Concurrent processing and data fusion. *Proceedings of the Conference on Human Factors in Computing Systems (INTERCHI '93),* pp 172.

NimbleVR. (2014). Retrieved from http://nimblevr.com.

Nvidia PhysX. (2015). Retrieved from http://developer.nvidia.com/physx.

Oculus. (2015). Retrieved from https://www.oculus.com.

Ohshima, T., Satoh, K., Yamamoto, H., & Tamura, H. (1998). AR2Hockey: A case study of collaborative augmented reality. *Proceedings of the IEEE Virtual Reality Annual International Symposium (VRAIS'98),* pp 268-275.

Oikonomidis, I., Kyriazis, N., & Argyros, A. A. (2011). Efficient model-based 3D tracking of hand articulations using Kinect. *Proceedings of the 22nd British Machine Vision Conference (BMVC'11).*

Oikonomidis, I., Kyriazis, N., & Argyros, A. A. (2012). Tracking the articulated motion of two strongly interacting hands. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, (CVPR'12),* .

Olwal, A., Benko, H., & Feiner, S. (2003). SenseShapes: using statistical geometry for object selection in a multimodal augmented reality. *Proceedings of The Second IEEE and ACM International Symposium on.Mixed and Augmented Reality (ISMAR'02),* pp 300-301.

Open Dynamic Engine. (2015). Retrieved from http://www.ode.org.

OpenGL. (2015). Retrieved from http://www.opengl.org.

OpenNI. (2015). Retrieved from http://structure.io/openni.

OpenSceneGraph. (2015). Retrieved from http://www.openscenegraph.org.

Oviatt, S. (1999). Ten myths of multimodal interaction. *Communications of the ACM, 42,* pp 74-81.

Oviatt, S. L., Cohen, P. R., Wu, L., Vergo, J., Duncan, L., Suhm, B., Bers, J., . . . Ferro, D. (2000). Designing the user interface for multimodal speech and gesture applications: State-of-the-art systems and research directions. *Human Computer Interaction, 15*(4), pp 263–322.

Perritaz, D., Salzmann, C., & Gillet, D. (2009). Quality of experience for adaptation in augmented reality. *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics. (SMC'09),* pp 888-893.

Piumsomboon, T., Clark, A., Billinghurst, M., & Cockburn, A. (2013). User-Defined Gestures for Augmented Reality. *In P. Kotzé, G. Marsden, G. Lindgaard, J. Wesson, & M. Winckler (Eds.), Human-Computer Interaction (INTERACT'13)*, Vol. 8118, pp. 282-299.

Piumsomboon, T., Clark, A., Umakatsu, A., & Billinghurst, M. (2012). Physically-based natural hand and tangible AR interaction for face-to-face collaboration on a tabletop. *The IEEE Symposium on 3D User Interfaces (3DUI'12),* pp 155-156.

PMD Technologies. (2014). Retrieved from http://www.pmdtec.com.

Poelman, R., Akman, O., Lukosch, S., & Jonker, P. (2012). As if being there: mediated reality for crime scene investigation. *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work (CSCW'12),* pp 1267-1276 .

Primesense. OpenNI. Retrieved from http://structure.io/openni.

Project Tango. Retrieved from https://www.google.com/atap/projecttango.

Reeves, L. M., Lai, J., Larson, J. A., Oviatt, S., Balaji, T. S., Buisine, S., & Wang, Q. (2004). Guidelines for multimodal user interface design. *Communications of the ACM, 47,* pp 57-59.

Regenbrecht, H. T., Wagner, M. T., & Baratoff, G. (2002). MagicMeeting: a collaborative tangible augmented reality system. *Virtual Reality, 6*(3), pp 151-166.

Reicher, T., Mac Williams, A., Brugge, B., & Klinker, G. (2003). Results of a study on software architectures for augmented reality systems. *Proceedings the Second IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'03)*, pp 274-275.

Rekimoto, J., & Saitoh, M. (1999). Augmented surfaces: a spatially continuous work space for hybrid computing environments. *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*, pp 378-385.

Rhienmora, P., Gajananan, K., Haddawy, P., Dailey, M. N., & Suebnukarn, S. (2010). Augmented reality haptics system for dental surgical skills training. *Proceeding of the 17th ACM Symposium on Virtual Reality Software and Technology (VRST'10)*, pp 97-98 .

Ross, E. (2000). *Intelligent User Interfaces: Survey and Research Directions*. Department of Computer Science, University of Bristol.

Ruiz, J., Li, Y., & Lank, E. (2011). User-defined motion gestures for mobile interaction. *Proceedings of the annual conference on Human factors in computing systems (CHI'11)*, pp 197-206.

Rusu, R. B., & Cousins, S. (2011). 3D is here: Point Cloud Library (PCL). *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'11)*, pp 1-4.

Schmalstieg, D., Fuhrmann, A., & Hesina, G. (2000). Bridging multiple user interface dimensions with augmented reality. *Proceedings of the IEEE and ACM International Symposium on the Augmented Reality (ISAR'00)*, pp 20-29.

Schmalstieg, D., Fuhrmann, A., Szalavari, Z., & Gervautz, M. (1996). Studierstube - An Environment for Collaboration in Augmented Reality. *Proceedings of CVE'96 Workshop*.

Schuler, D., & Namioka, A. (1993). *Participatory Design: Principles and Practices*. Hillsdale, NJ: Lawrence Erlbaum.

Scott, S. D., & Carpendale, S. (2010). Theory of Tabletop Territoriality. In C. Müller-Tomfelde (Ed.), *Tabletops - Horizontal Interactive Displays*, *Springer*, pp. 357-385.

Shneiderman, B., & Maes, P. (1997). Direct manipulation vs. interface agents. *interactions, 4,* pp 42-61.

Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., . . . Blake, A. (2011). Real-time human pose recognition in parts from single depth images. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'11),* pp 1297-1304.

Sodhi, R., Poupyrev, I., Glisson, M., and Israr, A. (2013). AIREAL: interactive tactile experiences in free air, *ACM Trans. Graph., 32,* (4), pp. 1-10.

SoftKinetic. Retrieved from http://www.softkinetic.com.

Song, P., Yu, H., & Winkler, S. (2008). Vision-based 3D finger interactions for mixed reality games with physics simulation. *Proceedings of the 7th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry (VRCAI'08).*

Structure Sensor. Retrieved from http://structure.io.

Sugano, N., Kato, H., & Tachibana, K. (2003). The effects of shadow representation of virtual objects in augmented reality. *Proceedings the Second IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'03),* pp 76-83.

Sutherland, I. E. (1964). Sketch pad a man-machine graphical communication system. *Proceedings of the SHARE design automation workshop*, pp 6.329-6.346.

Sutherland, I. E. (1965). The Ultimate Display. *Proceedings of the IFIP Congress*, pp 506-508.

Swan, J. E., & Gabbard, J. L. (2005). *Survey of User-Based Experimentation in Augmented Reality.* Paper presented at the Proceedings of 1st International Conference on Virtual Reality.

Tamura, H., Yamamoto, H., & Katayama, A. (2001). Mixed reality: future dreams seen at the border between real and virtual worlds. *Computer Graphics and Applications, IEEE, 21*(6), 64-70.

Tokamak Physics. (2015). Retrieved from http://www.tokamakphysics.com.

van Krevelen, D. W. F., & Poelman, R. (2010). A Survey of Augmented Reality Technologies, Applications and Limitations. *The International Journal of Virtual Reality*, *9*(2), pp 1-20.

Ventura, J., & Hollerer, T. (2009). Online environment model estimation for augmented reality. *Proceedings of the 8th IEEE International Symposium on Mixed and Augmented Reality (ISMAR'09)*, pp 103-106.

Wang, R., Paris, S., Popovic, J. (2011). 6D hands: markerless hand-tracking for computer aided design. *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pp 549-558.

Wang, Y., & Samaras, D. (2003). Estimation of multiple directional light sources for synthesis of augmented reality images. *Graphical Models*, *65*, pp 185-205.

Webb, J., & Ashley, J. (2012). *Beginning Kinect Programming with the Microsoft Kinect SDK*: Apress.

Weiser, M. (1993). Hot topics-ubiquitous computing. *Computer*, *26*, pp 71-72.

Wilson, A. D. (2007). Depth-Sensing Video Cameras for 3D Tangible Tabletop Interaction. *Proceedings of the Second Annual IEEE International Workshop on Horizontal Interactive Human-Computer Systems (TABLETOP'07)*, 201-204.

Wilson, A. D. (2010). Using a depth camera as a touch sensor. *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces (ITS'10)*, pp 69-72.

Wilson, A. D., & Benko, H. (2010). Combining multiple depth cameras and projectors for interactions on, above and between surfaces. Proceedings of the 23nd annual ACM symposium on User interface software and technology (UIST'10), pp 273-282.

Wilson, A. D., Izadi, S., Hilliges, O., Garcia-Mendoza, A., & Kirk, D. (2008). Bringing physics to the surface. *Proceedings of the 21st annual ACM symposium on User interface software and technology (UIST'08)*, pp 67-76.

Wobbrock, J. O., Aung, H. H., Rothrock, B., & Myers, B. A. (2005). Maximizing the guessability of symbolic input. *Proceedings of the Extended abstracts on Human factors in computing systems (CHI'05)*, pp 1869-1872.

Wobbrock, J. O., Morris, M. R., & Wilson, A. D. (2009). User-defined gestures for surface computing. *Proceedings of the 27th international conference on Human factors in computing systems (CHI'09)*, pp 1083-1092.

Wobbrock, J. O., Wilson, A. D., & Li, Y. (2007). Gestures without libraries, toolkits or training: a $1 recognizer for user interface prototypes. *Proceedings of the 20th annual ACM symposium on User interface software and technology (UIST'07)*, pp 159-168.

Xiao, R., Harrison, C., & Hudson, S. E. (2013). WorldKit: rapid and easy creation of ad-hoc interactive applications on everyday surfaces. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'13)*, pp 879-888.

Zhang, Z. (2012). Microsoft Kinect Sensor and Its Effect. *IEEE MultiMedia*, *19*(2), pp 4-10.

Zhou, F., Dun, H. B.-L., & Billinghurst, M. (2008). Trends in augmented reality tracking, interaction and display: A review of ten years of ISMAR. *Proceeding of the 7th IEEE International Symposium on Mixed and Augmented Reality (ISMAR'08)*, pp 193-202.

Zhu, J., Pan, Z., Sun, C., & Chen, W. (2010). Handling occlusions in video-based augmented reality using depth information. *Computer Animation and Virtual Worlds, 21*(Compendex), pp 509-521.

Appendices

Appendix A

User-defined Gestures Study Material

The following material from the guessability study in Chapter 5 is presented on the following pages:

1. The information sheet provided to participants before they agreed to participate in the study.

2. The consent form that was signed by all participants before participating in the study.

3. The pre-experiment and post-experiment questionnaires.

Department of Computer Science and Software Engineering

UNIVERSITY OF CANTERBURY
Te Whare Wānanga o Waitaha
CHRISTCHURCH NEW ZEALAND

Professor Andy Cockburn
Tel: +64 3 364 2987 x7768, Fax: + 64 364 2569
Email:andy@cosc.canterbury.ac.nz

Thammathip Piumsomboon
Tel: +64 21 1595734
Email:thammathip.piumsomboon@pg.canterbury.ac.nz

Information Sheet for "User-Defined Gestures for Augmented Reality"

You are invited to participate in the research project "User-Defined Gesture for Augmented Reality".

The aim of this study

The aims of this research are (1) to elicit gestures from the users and build the user-defined gestures set based on the consensus found, (2) to create taxonomy of gestures for Augmented Reality (AR), (3) to draw qualitative findings from think-aloud protocol and the interview through video analysis.

Who are the reserchers?

The researchers are:

- Thammathip Piumsomboon, Doctoral student.
- Prof. Andy Cockburn, Supervisor, Department of Computer Science and Software Engineering.
- Prof. Mark Billinghurst, Co-supervisor, Human Interface Laboratory (HITLabNZ).
- Dr. Adrian Clark, Advisor, Human Interface Laboratory (HITLabNZ).

How are participants selected for this study?

Volunteer undergraduate and postgraduate students are invited to participate in the experiments through online notices and emails. We offer $10 café voucher that serves as a reward for participation.

What will the research involve?

In the experiment, the participants will be wearing a head mounted display (HMD) to watch the simulated effect of the AR tasks. They will be asked to design hand gestures that they think evoke such effects. There are thirty-two tasks under five categories that

should take 80 minutes or less to complete. After each gesture, they are asked to give it a score on a 7-point Likert scale on goodness and ease. Finally, a brief interview will be conducted that take 10 minutes or less to complete, so that the participants can comment on the experiment and suggest the possible tasks and corresponding gestures that is not included.

What are the benefits of the study?

This study will give participants the opportunity to experience the AR technology. In turn, participants will contribute to enriching knowledge in the area of gesture interaction in AR by designing a set of hand gestures.

Do I have to take part?

No, your participation is entirely voluntary and you also have the right to withdraw from the project at any time, including withdrawal of any information provided.

What will happen to the results of this study and will my information in this study be kept confidential?

The results of the project may be published (including in a publicly accessible PhD thesis), but you are assured of the complete confidentiality of data gathered in this investigation: the identity of participants will not be made public. To ensure anonymity and confidentiality your signed consent form will be stored in a locked filing cabinet in a locked office, and computer logs of your participation are anonymous.

Who has approved this study?

The project has been reviewed ***and approved*** by the University of Canterbury Human Ethics Committee.

Please contact *Thammathip Piumsomboon* if you have further questions.

Department of Computer Science and Software Engineering

UNIVERSITY OF CANTERBURY
Te Whare Wānanga o Waitaha
CHRISTCHURCH NEW ZEALAND

Professor Andy Cockburn
Tel: +64 3 364 2987 x7768, Fax: + 64 364 2569
Email:andy@cosc.canterbury.ac.nz

Thammathip Piumsomboon
Tel: +64 21 1595734
Email:thammathip.piumsomboon@pg.canterbury.ac.nz

August, 2012

Consent form for "User-Defined Gestures for Augmented Reality"

I have read and understood the description of the above-named project. On this basis I agree to participate as a subject in the project, and I consent to publication of the results of the project with the understanding that anonymity will be preserved. I am over 18 years of age.

I understand also that I may at any time withdraw from the project, including withdrawal of any information I have provided.

I note that the project has been reviewed ***and approved*** by the University of Canterbury Human Ethics Committee.

NAME (please print): ……

Signature:

Date:

Pre-experiment questionnaire

What is your Participant Number?

What's your gender?

- Male
- Female

How old are you?

Are you left-handed or right-handed?

- Right-handed
- Left-handed
- I can use both hands very well.

Have you had any experience with Augmented Reality? (e.g. mobile phone, Google Glass, HMD etc.)

- Yes
- No

On which interfaces or devices that you had the Augmented Reality experience?

If you have some experience in Augmented Reality, at what level of experience do you consider yourself to be?

- Beginner (little knowledge on the topic but had some experience)
- Intermediate (Know about it and had some experience)
- Knowledgeable (Know it and use it regularly)
- Expert (Very adept in the topic and use it all the time)

Post-experiment questionnaire

Description: Each participant will be requested to complete the same two 7-point Likert scales questions for all thirty-two tasks, as shown in table 1, after designing the gesture for each task.

Table 1: The thirty-two tasks that the participant will be designing the gestures for.

Category		Tasks	Category	Tasks
Transform	*Move*	1. Short distance	*Editing*	17. Single selection
		2. Long distance		18. Multiple selection
	Rotation	3. Pitch (y-axis)		19. Insert
		4. Roll (x-axis)		20. Delete
		5. Yaw (z-axis)		21. Undo
	Scale	6. Uniform scale		22. Redo
		7. X-axis		23. Group
		8. Y-axis		24. Ungroup
		9. Z-axis		25. Accept
Simulation		10. Play		26. Reject
		11. Pause		27. Copy
		12. Stop (Reset)		28. Cut
		13. Increase speed		29. Paste
		14. Decrease speed	*Menu*	30. Open menu
Browsing		15. Previous		31. Close menu
		16. Next		32. Select an option

Please respond to the following statements by placing a tick in the corresponding circle.

	Score						
This gesture is a good match for performing this task.	○ 1 Strongly Disagree	○ 2	○ 3	○ 4	○ 5	○ 6	○ 7 Strongly Agree
This gesture is easy to perform.	○ 1 Strongly Disagree	○ 2	○ 3	○ 4	○ 5	○ 6	○ 7 Strongly Agree

Please write any comments on the given gesture below:

Appendix B

Usability Study Material

The following material from the comparison study between direct and indirect natural interaction techniques in Chapter 8 is presented on the following pages:

1. The information sheet provided to participants before they agreed to participate in the study.

2. The consent form that was signed by all participants before participating in the study.

3. The pre-experiment questionnaires.

4. The post-experiment questionnaires.

Professor Mark Billinghurst
Email: mark.billinghurst@canterbury.ac.nz

Thammathip Piumsomboon
Telephone: +64 3 364 2349
Email:thammathip.piumsomboon@pg.canterbury.ac.nz
Date:

Information Sheet for "A Comparison of Direct and Indirect Natural Interaction Techniques in Augmented Reality"

You are invited to take part in an Augmented Reality interface and interaction research study. Before you decide to be part of this study, you need to understand the risks and benefits. This information sheet provides information about this research study. A researcher will be available to answer your questions and provide further explanations. If you agree to take part in the research study, you will be asked to sign to the consent form.

The aim of this study is to gain your impression interacting with our Augmented Reality (AR) interface and ask you to evaluate two natural interaction techniques performing the given tasks in an AR environment.

The researchers are:
Thammathip Piumsomboon, PhD candidate.
Prof. Mark Billinghurst, Project Supervisor, Human Interface Laboratory New Zealand (HITLabNZ).

Keywords:
Augmented Reality (AR) – technology that overlays computer graphics on to the real world.
Gesture Interface - an interface that use hand gestures as a computer input.
Gesture-speech/multimodal Interface – an interface that use hand gestures and speech as computer inputs.

What will the research involve?
You will be wearing a video see-through head mounted display (HMD). Through the HMD, you will be able to see the real-world with superimposed graphics in 3D. There are two interaction techniques, Grasp-Shell and Gesture-Speech, for you to perform in three tasks, which are single object relocation, multiple object relocation, and uniform resizing. For Grasp-Shell, you will be able to grasp and manipulate virtual 3D objects with your bare hands. With Gesture-Speech, you will be able to use hand pointing and speech command to manipulate those virtual objects. Upon each task completion, you will be asked to evaluate the usability of each interaction technique for performing each task through the online questionnaire. A video will be recorded during the experiment while you are interacting with the system. The study will take approximately 90 minutes or less including 5 minutes break between each task.

What is the procedure?
You will be asked to complete a pre-questionnaire that assess your experience with AR interface, gesture interface, and multimodal interface. You will be given 5 minutes to learn to use each interaction technique and another 5 minutes in a practice session that will be similar to the tasks in this experiment. During this learning period, the system will be calibrated so that it works well for you. This calibration involves adjusting parameters to ensure accurate hand pointing regardless of hand size, and selection of verbal commands that the speech recognizer could accurately determine for you.

There are three tasks for you to perform. You will be asked to perform both interaction techniques for the current task before advancing to the next task. Task 1 focused on moving a single virtual object to the target, and featured ten subtasks. Task 2 focused on moving three objects comprising of five subtasks involving moving without rotating the object, and moving with rotation. Task 3 focused on resizing objects of varying shapes and sizes and was comprised of five subtasks.

At the beginning of each task, you will have a 3 second countdown, which will be displayed in the center of your view. As each task begins and for every successful target matches, a sound of a bell will be played. The object will be displayed with opaque textures and a red outline, while the target will be 50% translucent with a yellow outline. The task completion time will be started after the bell rang and stopped when all the targets in the scene are matched.

The questionnaire will be conducted online with the Qualtrics survey. The overall time of the experiment should be less than 90 minutes. After each task completion for each interaction technique, you will be asked to complete a usability questionnaire and at the completion of each task for both interaction techniques, you will also need to complete a post-task questionnaire.

What are the potential risks or discomforts in the study?
Risks are minimal in this study. You will be wearing a video see-through head mounted display (HMD). The researcher will make sure that you are comfortable wearing the HMD and during usage and do not suffer from a simulator sickness that might cause eyestrain problems, nausea, headaches etc. If the simulation sickness does occur, the researcher will immediately terminate the experiment and still offer the voucher as a reward to you for participating.

What are the benefits of the study?
This study will give you an opportunity to experience a cutting-edge AR interface and interaction. In turn, you will be helping us in improving our interface and interaction techniques
and our findings will benefit the research community and in turn the development of the future interface.

Do I have to take part?
Participation is voluntary and you have the right to withdraw at any stage without penalty. If you withdraw, we will remove information relating to you. You may receive a copy of the project results by contacting the researcher at the conclusion of the project.

What will happen to the results of this study and will my information in this study be kept confidential?
The results of the project may be published (including in a publicly accessible PhD thesis), but you are assured of the complete confidentiality of data gathered in this investigation: your identity will not be made public. To ensure anonymity and confidentiality your signed consent form will be stored in a locked filing cabinet in a locked office, and computer logs of your participation will be anonymous. All data collected for the study will be destroyed after ten years. A thesis is a public document and will be available through the UC Library.

Will I receive any compensation?
You will receive a $10NZ voucher for your participation.

The project is being carried out as part of a PhD program by *Thammathip Piumsomboon* under the supervision of *Prof.Mark Billinghurst*, who can be contacted at mark.billinghurst@canterbury.ac.nz. He will be pleased to discuss any concerns you may have about participation in the project.

This project has been reviewed and approved by the University of Canterbury Human Ethics Committee, and participants should address any complaints to The Chair, Human Ethics Committee, University of Canterbury, Private Bag 4800, Christchurch (human-ethics@canterbury.ac.nz).

If you agree to participate in the study, you are asked to complete the consent form and return to the researcher.

Professor Mark Billinghurst
Email: mark.billinghurst@canterbury.ac.nz
Thammathip Piumsomboon
Telephone: +64 3 364 2349
Email:thammathip.piumsomboon@pg.canterbury.ac.nz
Date:

Consent Form for "A Comparison of Direct and Indirect Natural Interaction Techniques in Augmented Reality"

I have been given a full explanation of this project and have had the opportunity to ask questions.

I understand what is required of me if I agree to take part in the research.

I understand that participation is voluntary and I may withdraw at any time without penalty. Withdrawal of participation will also include the withdrawal of any information I have provided should this remain practically achievable.

I understand that any information or opinions I provide will be kept confidential to the researcher and that any published or reported results will not identify the participants. I understand that a thesis is a public document and will be available through the UC Library.

I understand that all data collected for the study will be kept in locked and secure facilities and/or in password protected electronic form and will be destroyed after ten years.

I understand the risks associated with taking part and how they will be managed.

I understand that I am able to receive a report on the findings of the study by contacting the researcher at the conclusion of the project.

I understand that I can contact the researcher, Thammathip Piumsomboon (thammathip.piumsomboon@pg.canterbury.ac.nz), or supervisor, Prof. Mark Billinghurst (mark.billinghurst@canterbury.ac.nz), for further information.

This project has been reviewed and approved by the UC Human Ethics Committee (8.1 UC HEC Policy). If I have any complaints, I can contact the Chair of the University of Canterbury Human Ethics Committee, Private Bag 4800, Christchurch (human-ethics@canterbury.ac.nz)

By signing below, I agree to participate in this research project.

_____	_____	_____
Participant (Print name)	Signature	Date

Pre-experiment questionnaire

What is your Participant Number?

What's your gender?

- Male
- Female

How old are you?

Are you left-handed or right-handed?

- Right-handed
- Left-handed
- I can use both hands very well.

How proficient is your English skill in speaking?

- Beginner
- Intermediate
- Fluent
- Native speaker

Have you had any experience with Augmented Reality? (e.g. mobile phone, Google Glass, HMD etc.)

- Yes
- No

On which interfaces or devices that you had the Augmented Reality experience?

If you have some experience in Augmented Reality, at what level of experience do you consider yourself to be?

- Beginner (little knowledge on the topic but had some experience)
- Intermediate (Know about it and had some experience)
- Knowledgeable (Know it and use it regularly)
- Expert (Very adept in the topic and use it all the time)

Have you had any experience with Gesture Interface? (such as the Kinect, Wii)

- Yes
- No

On which interfaces or devices that you had the Gesture Interface experience?

If you have some experience in Gesture Interface, at what level of experience do you consider yourself to be?

- Beginner (little knowledge on the topic but had some experience)
- Intermediate (Know about it and had some experience)
- Knowledgeable (Know it and use it regularly)
- Expert (Very adept in the topic and use it all the time)

Have you had any experience with Gesture and Speech Interface? (e.g. XBox One + new Kinect, Siri on IOS etc)

- Yes
- No

On which interfaces or devices that you had the Gesture Interface experience?

If you have some experience in Gesture and Speech Interface, at what level of experience do you consider yourself to be?

- Beginner (little knowledge on the topic but had some experience)
- Intermediate (Know about it and had some experience)
- Knowledgeable (Know it and use it regularly)
- Expert (Very adept in the topic and use it all the time)

Please complete the experiment before proceeding with the next questionnaire.

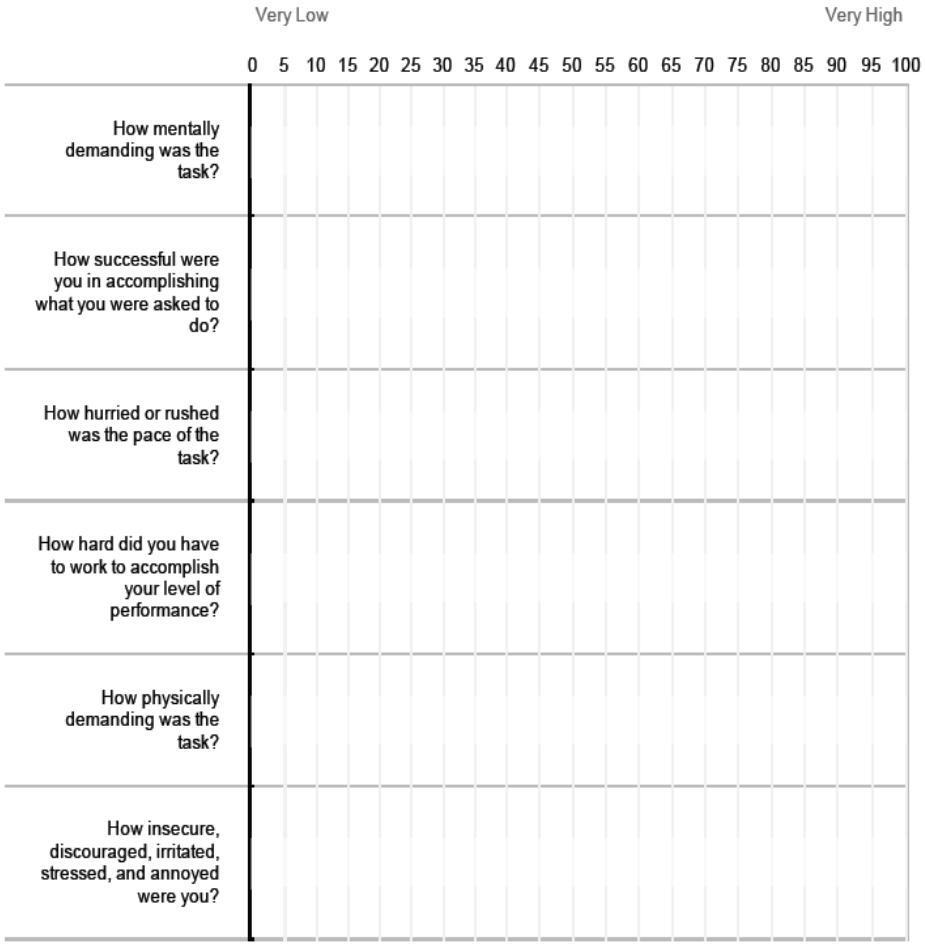
Questionnaire after task completion for each interaction technique (Same for all)

Task 1: SINGLE MOVE
Condition 1

Which Interface did you use?

Grasp-Shell ○ Gesture-Speech ○

Please rate



Please rate the system interaction according to the following attributes of usability

	Strongly Disagree			Neutral			Strongly Agree
Naturalness - It was natural to perform the task with this Interface.	○	○	○	○	○	○	○
Intuitiveness - It was intuitive to perform the task with this Interface.	○	○	○	○	○	○	○
Fun - It was fun to perform the task with this interface.	○	○	○	○	○	○	○
Efficiency - It was efficient to complete the task with this interface.	○	○	○	○	○	○	○
Accuracy - It was accurate to perform the task with this interface.	○	○	○	○	○	○	○
Learnability - It was easy to learn how to use the interface.	○	○	○	○	○	○	○
Memorability - It was easy to remember how to use the interface.	○	○	○	○	○	○	○
Satisfaction - It was satisfying performing the task with this interface.	○	○	○	○	○	○	○

What do you like or dislike about this interface when performing the given task?

From the above question, why do you like or dislike it?

Post-task questionnaire (Same for all tasks)

Please complete the experiment before proceeding with the next questionnaire.

Which interfaces do you like better to perform this task?

- ○ Grasp-Shell
- ○ Gesture-Speech
- ○ The same

In this section there are a number of statements concerning the reasons why you prefer one interface over another in the previous question.

Please read each statement carefully and select whether or not each statement is TRUE for you personally. Base your answers on your experience of the previous interacting conditions.

When performing the same task, I would prefer the interface I chose because...

	Not at all true for me						Very true for me
It was more accurate (I can do the task with fewer errors)	○	○	○	○	○	○	○
It was more intuitive.	○	○	○	○	○	○	○
It was more satisfying to use.	○	○	○	○	○	○	○
It was easier to remember the commands	○	○	○	○	○	○	○
It was more efficient.	○	○	○	○	○	○	○
It was easier to learn.	○	○	○	○	○	○	○
It was more natural.	○	○	○	○	○	○	○
It was more fun and enjoyable.	○	○	○	○	○	○	○

Other reasons that can explain your preference of one condition over the others: